Beyond Bases: Frame Theory and Applications

Mark Magsino

The Ohio State University

U.S. Naval Academy
December 9, 2021

Table of Contents

Table of Contents

Frames are bases with more vectors.

- A basis for $\mathbb{R}^d$ is a linearly independent spanning set of d vectors.
- A frame for $\mathbb{R}^d$ is **any** spanning set of $n \geq d$ vectors.
- We can think of them as bases with some extra vectors added.
- Extra vectors $\rightarrow$ no linear independence and redundant
- Linear independence $\rightarrow$ unique representations

Frames are bases with more vectors.

- A basis for $\mathbb{R}^d$ is a linearly independent spanning set of d vectors.
- A frame for $\mathbb{R}^d$ is **any** spanning set of $n \geq d$ vectors.
- We can think of them as bases with some extra vectors added.
- Extra vectors $\rightarrow$ no linear independence and redundant
- Linear independence $\rightarrow$ unique representations

If frames are redundant and linear independence is nice, **why use frames?**

Redundancy combats data corruption.

Suppose we want to send the vector $(1, 2, 3)$.

Redundancy combats data corruption.

Suppose we want to send the vector $(1, 2, 3)$.

- What if one of the coefficients disappears? $(1, ?, 3)$

Redundancy combats data corruption.

Suppose we want to send the vector $(1, 2, 3)$.

- What if one of the coefficients disappears? $(1, ?, 3)$
- If we send 3 copies of each coefficient, it's not as bad if one goes missing.
  $(1, 1, 1 \mid 2, 2, 2 \mid 3, 3, 3) \to (1, 1, 1 \mid 2, ?, 2 \mid 3, 3, 3)$

Redundancy combats data corruption.

Suppose we want to send the vector $(1, 2, 3)$.

- What if one of the coefficients disappears? $(1, ?, 3)$
- If we send 3 copies of each coefficient, it's not as bad if one goes missing.
  $(1, 1, 1 \mid 2, 2, 2 \mid 3, 3, 3) \to (1, 1, 1 \mid 2, ?, 2 \mid 3, 3, 3)$
- Real World Example: (3,1)-repetition code

Redundancy combats data corruption.

Suppose we want to send the vector $(1, 2, 3)$.

- What if one of the coefficients disappears? $(1, ?, 3)$
- If we send 3 copies of each coefficient, it's not as bad if one goes missing.
  $(1, 1, 1 \mid 2, 2, 2 \mid 3, 3, 3) \to (1, 1, 1 \mid 2, ?, 2 \mid 3, 3, 3)$
- Real World Example: (3,1)-repetition code

Redundant frames help ease losses in storage or transmission. What else can they do?

Frames allow for efficient data storage.

Suppose we want to store the vector $(1, 2, 3, 4, 5)$.

Frames allow for efficient data storage.

Suppose we want to store the vector $(1, 2, 3, 4, 5)$.

- Requires 5 coefficients; one for each coordinate.

Frames allow for efficient data storage.

Suppose we want to store the vector $(1, 2, 3, 4, 5)$.

- Requires 5 coefficients; one for each coordinate.
- What if we add a 6th vector to basis: $(1, 2, 3, 4, 5)$.
  $\rightarrow$ Can reconstruct w/ coefficients $(0, 0, 0, 0, 0, 1)$.

Frames allow for efficient data storage.

Suppose we want to store the vector $(1, 2, 3, 4, 5)$.

- Requires 5 coefficients; one for each coordinate.
- What if we add a 6th vector to basis: $(1, 2, 3, 4, 5)$.
  $\rightarrow$ Can reconstruct w/ coefficients $(0, 0, 0, 0, 0, 1)$.
- Store as $(0, 5, 1)$; next number after a zero is $\#$ of zeros.

Frames allow for efficient data storage.

Suppose we want to store the vector $(1, 2, 3, 4, 5)$.

- Requires 5 coefficients; one for each coordinate.
- What if we add a 6th vector to basis: $(1, 2, 3, 4, 5)$.
  $\rightarrow$ Can reconstruct w/ coefficients $(0, 0, 0, 0, 0, 1)$.
- Store as $(0, 5, 1)$; next number after a zero is # of zeros.
- Real World Example: JPEG image compression

Frames allow for efficient data storage.

Suppose we want to store the vector $(1, 2, 3, 4, 5)$.

- Requires 5 coefficients; one for each coordinate.
- What if we add a 6th vector to basis: $(1, 2, 3, 4, 5)$.
  $\rightarrow$ Can reconstruct w/ coefficients $(0, 0, 0, 0, 0, 1)$.
- Store as $(0, 5, 1)$; next number after a zero is # of zeros.
- Real World Example: JPEG image compression

Frames can help compress data. How do we know which frames are "good" for representing data?

Orthonormal bases makes vector representation easy.

- A set of vectors $\{v_j\}_{j=1}^d$ is spanning if for any vector $x \in \mathbb{R}^d$ there exist coefficients c_j such that

$$x = \sum_{j=1}^{d} c_j v_j$$

- If $\{v_j\}_{j=1}^d$ is an **orthonormal basis** then for every vector $x \in \mathbb{R}^d$,

$$x = \sum_{j=1}^{d} (v_j \cdot x) v_j.$$

Orthonormal bases makes vector representation easy.

- A set of vectors $\{v_j\}_{j=1}^d$ is spanning if for any vector $x \in \mathbb{R}^d$ there exist coefficients c_j such that

$$x = \sum_{j=1}^{d} c_j v_j$$

- If $\{v_j\}_{j=1}^d$ is an **orthonormal basis** then for every vector $x \in \mathbb{R}^d$,

$$x = \sum_{j=1}^{d} (v_j \cdot x) v_j.$$

When we have orthonormal bases, reconstruction coefficients are easy to compute! Do have have this kind of formula for frames?

Frames also have a reconstruction formula.

Theorem

Let $\{v_j\}_{j=1}^n$ *be a frame for* $\mathbb{R}^d$ *and let* $\mathbf{S}$ *be its frame operator. Then,*

$$x = \sum_{j=1}^{n} (\mathbf{S}^{-1} v_j \cdot x) v_j, \quad \forall x \in \mathbb{R}^d.$$

- Frame operators are linear mappings $\rightarrow$ it has a matrix form!
- $\mathbf{S}$ is always invertible but matrix inversion can be expensive.
- For orthonormal bases $\mathbf{S} = \mathbf{I}$.
- Other reconstruction coefficients possible!

Frames also have a reconstruction formula.

Theorem

Let $\{v_j\}_{j=1}^n$ *be a frame for* $\mathbb{R}^d$ *and let* $\mathbf{S}$ *be its frame operator. Then,*

$$x = \sum_{j=1}^{n} (\mathbf{S}^{-1} v_j \cdot x) v_j, \quad \forall x \in \mathbb{R}^d.$$

- Frame operators are linear mappings $\rightarrow$ it has a matrix form!
- $\mathbf{S}$ is always invertible but matrix inversion can be expensive.
- For orthonormal bases $\mathbf{S} = \mathbf{I}$.
- Other reconstruction coefficients possible!

The inverse of frame operators are needed for the reconstruction formula, but it can be hard to compute. Can we make it easier?

Tight frames give us good frame operators.

- Let $\{v_j\}_{j=1}^n$ be a frame for $\mathbb{R}^d$. The frame operator, $\mathbf{S} : \mathbb{R}^d \to \mathbb{R}^d$, is the linear map
$$\mathbf{S}(x) = \sum_{j=1}^{n} (v_j \cdot x) v_j.$$
- $\mathbf{S}$ is symmetric and positive definite $\rightarrow$ $\mathbf{S}$ has real and positive eigenvalues.
- A frame is tight if all those eigenvalues are the same.
- In this case we have $\mathbf{S} = A\mathbf{I}$, where A is the repeated eigenvalue.

Tight frames give us good frame operators.

- Let $\{v_j\}_{j=1}^n$ be a frame for $\mathbb{R}^d$. The frame operator, $\mathbf{S} : \mathbb{R}^d \to \mathbb{R}^d$, is the linear map
$$\mathbf{S}(x) = \sum_{j=1}^{n} (v_j \cdot x) v_j.$$
- $\mathbf{S}$ is symmetric and positive definite $\to$ $\mathbf{S}$ has real and positive eigenvalues.
- A frame is tight if all those eigenvalues are the same.
- In this case we have $\mathbf{S} = A\mathbf{I}$, where A is the repeated eigenvalue.

Tight frames are good for computing. Can we get more? Can we get something like orthogonality?

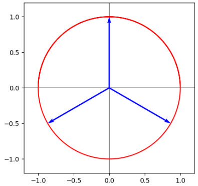
Equiangular is the next closest thing to orthogonal.

A set of unit vectors $\{v_j\}_{j=1}^n$ in $\mathbb{R}^d$ is an equiangular tight frame (ETF) for $\mathbb{R}^d$ if:

1. $|v_j \cdot v_k|$ is constant across all $j \neq k$, and
2. $\{v_j\}_{j=1}^n$ is a tight frame.

Do they always exist?

- For $\mathbb{R}^d$, $n \leq d(d+1)/2$ is required.
- Even if $n \leq d(d+1)/2$, sometimes ETFs of n vectors in $\mathbb{R}^d$ don't exist!
- There is an ETF of $d+1$ vectors in $\mathbb{R}^d$.



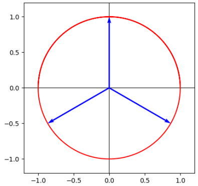
Equiangular is the next closest thing to orthogonal.

A set of unit vectors $\{v_j\}_{j=1}^n$ in $\mathbb{R}^d$ is an equiangular tight frame (ETF) for $\mathbb{R}^d$ if:

1. $|v_j \cdot v_k|$ is constant across all $j \neq k$, and
2. $\{v_j\}_{j=1}^n$ is a tight frame.

Do they always exist?

- For $\mathbb{R}^d$, $n \leq d(d+1)/2$ is required.
- Even if $n \leq d(d+1)/2$, sometimes ETFs of n vectors in $\mathbb{R}^d$ don't exist!
- There is an ETF of $d+1$ vectors in $\mathbb{R}^d$.



Equiangular tight frames are the frame version of orthonormal bases!

What have we learned about frames?

- Redundancy helps mitigate losses in storage and transmission.
- Frames can compress data.
- Tight frames make coefficients easy to compute.
- Equiangular tight frames are the next best thing to orthonormal bases.

What have we learned about frames?

- Redundancy helps mitigate losses in storage and transmission.
- Frames can compress data.
- Tight frames make coefficients easy to compute.
- Equiangular tight frames are the next best thing to orthonormal bases.

How do we use frames in practice?

Table of Contents

We can cheat when sampling low-dimensional signals.

Here's a scenario:

- A 10 year old needs a full body MRI scan.
- They can take up to **90 minutes**...
- What if we could shorten it by sampling less?
- **Underdetermined** system $\Rightarrow$ Non-unique solutions...

We can cheat when sampling low-dimensional signals.

Here's a scenario:

- A 10 year old needs a full body MRI scan.
- They can take up to **90 minutes**...
- What if we could shorten it by sampling less?
- **Underdetermined** system $\Rightarrow$ Non-unique solutions...
- Surprise: If target has **many zeros**, then it has a "unique" solution!

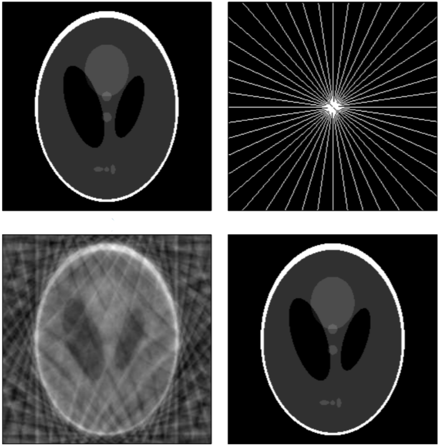
We can cheat when sampling low-dimensional signals.

Here's a scenario:

- A 10 year old needs a full body MRI scan.
- They can take up to **90 minutes**...
- What if we could shorten it by sampling less?
- **Underdetermined** system $\Rightarrow$ Non-unique solutions...
- Surprise: If target has **many zeros**, then it has a "unique" solution!

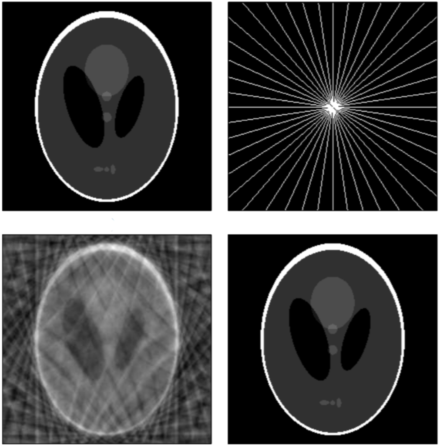
If our target signal is mostly zero, we can cheat and collect less! Where did this idea come from?

MRI scans are low-dimensional signals. [1]



[1] Figure 1 in Candes, Romberg, Tao 2005

MRI scans are low-dimensional signals. [1]



If we sample images with lots of zeros correctly, we can rebuild them near perfectly, or sometimes perfectly! Why does this work?

[1] Figure 1 in Candes, Romberg, Tao 2005

We can formulate the problem using frames.

- We wish to measure an image with n pixels using $d \ll n$ linear measurements.
- How do we recover the image from our undersampled measurements?

Problem

Let $d \ll n$. Given an $d \times n$ matrix $\mathbf{A}$ and $y \in \mathbb{R}^d$, solve

$$y = \mathbf{A}x.$$

We can formulate the problem using frames.

- We wish to measure an image with n pixels using $d \ll n$ linear measurements.
- How do we recover the image from our undersampled measurements?

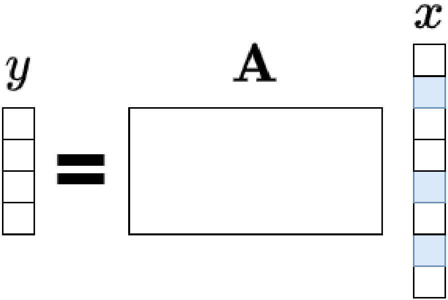
Problem

Let $d \ll n$. Given an $d \times n$ matrix $\mathbf{A}$ and $y \in \mathbb{R}^d$, solve

$$y = \mathbf{A}x.$$

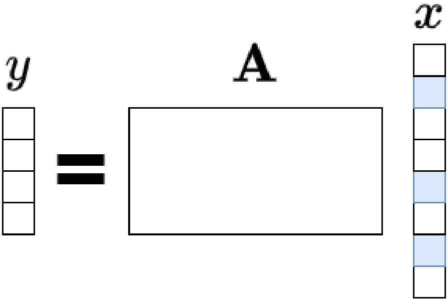
We are trying to solve a matrix-vector equation that should have many solutions. Why would having lots of zeros make that solution easier to find?

Solutions with many zeros have few variables.



- x is secretly a vector of 3 variables.
- 4 equations, 3 unknowns, could be solvable!
- We only know y...which x_i are the variables?

Solutions with many zeros have few variables.



- x is secretly a vector of 3 variables.
- 4 equations, 3 unknowns, could be solvable!
- We only know y...which x_i are the variables?

Our image secretly only has a few variables to solve for but which ones? How do we design $\mathbf{A}$ to account for this?

RIP accounts for different variable combinations.

Definition

Let k be a positive integer and $\delta > 0$. An $d \times n$ matrix $\mathbf{A}$ satisfies the (k, δ)-restricted isometry property (RIP) if for every vector $x \in \mathbb{R}^n$ with at most k nonzero entries

$$(1 - \delta)\|x\|^2 \leq \|\mathbf{A}x\|^2 \leq (1 + \delta)\|x\|^2.$$

- We call x with this property k-sparse.
- Mapping $\mathbf{A}$ to k-dimensional subspaces is almost an isometry $\rightarrow$ almost information preserving.
- This requires $\binom{n}{k}$ checks. This is computationally infeasible...

RIP accounts for different variable combinations.

Definition

Let k be a positive integer and $\delta > 0$. An $d \times n$ matrix $\mathbf{A}$ satisfies the (k, δ)-restricted isometry property (RIP) if for every vector $x \in \mathbb{R}^n$ with at most k nonzero entries

$$(1-\delta)\|x\|^2 \leq \|\mathbf{A}x\|^2 \leq (1+\delta)\|x\|^2.$$

- We call x with this property k-sparse.
- Mapping $\mathbf{A}$ to k-dimensional subspaces is almost an isometry $\rightarrow$ almost information preserving.
- This requires $\binom{n}{k}$ checks. This is computationally infeasible...

RIP matrices scan k-subspaces nicely, but they're hard to find. Is it worth it?

RIP quantifies how many zeros is enough.

Theorem (Candes '08)

Let k be a positive integer and let $\delta < \sqrt{2} - 1$. Suppose x' is a k-sparse vector we wish to recover. If $\mathbf{A}$ is $(2k, \delta)$-RIP, then the solution to

$$\operatorname{argmin} \quad \|x\|_1 \quad \textit{subject to} \quad y = \mathbf{A}x$$

is precisely x'. In particular, the recovery of x' is exact.

RIP quantifies how many zeros is enough.

Theorem (Candes '08)

Let k be a positive integer and let $\delta < \sqrt{2} - 1$. Suppose x' is a k-sparse vector we wish to recover. If $\mathbf{A}$ is $(2k, \delta)$-RIP, then the solution to

$$\operatorname{argmin} \quad \|x\|_1 \quad \textit{subject to} \quad y = \mathbf{A}x$$

is precisely x'. In particular, the recovery of x' is exact.

RIP matrices are good measurement matrices. If they're hard to check how do people find them?

Dealing with RIP is a battle between average and worst case performance.

How do people make RIP matrices?

1. Random matrices: Works with high probability but you can't be sure...
2. Explicit constructions: Can make some guarantees based on dot products between columns but worse performance than random.
3. Random subensembles: Explicit constructions but look at how random submatrices behave.

Dealing with RIP is a battle between average and worst case performance.

How do people make RIP matrices?

1. Random matrices: Works with high probability but you can't be sure...
2. Explicit constructions: Can make some guarantees based on dot products between columns but worse performance than random.
3. Random subensembles: Explicit constructions but look at how random submatrices behave.

Dealing with RIP is a battle between average and worst case performance.

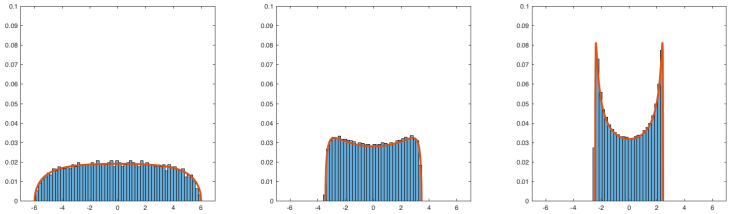
How do people make RIP matrices?

1. Random matrices: Works with high probability but you can't be sure...
2. Explicit constructions: Can make some guarantees based on dot products between columns but worse performance than random.
3. Random subensembles: Explicit constructions but look at how random submatrices behave.

Random ensembles balance explicit constructions and good average vs worst case performance.

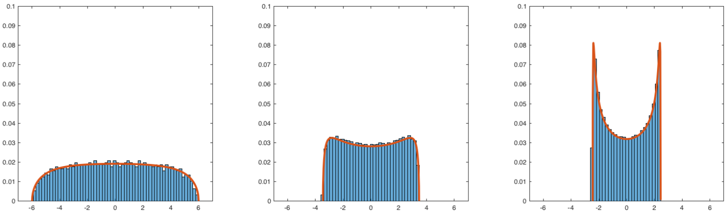
Paley ETFs exhibit a RIP-like behavior.

- Paley ETFs $\to$ Paley conference matrices. (Size: 10009×10009)
- For each $p \in \{0.1, 0.25, 0.4\}$ we draw a subset of indices $\mathcal{I}$ from $\{1, \cdots, 10009\}$ including each index independently with probability p.
- Take the principal submatrix using $\mathcal{I}$ as the subindices.
- The histograms of the eigenvalues of the submatrices appear to form a Kesten–McKay distribution with parameter $v = 1/p$!



Paley ETFs exhibit a RIP-like behavior.

- Paley ETFs $\rightarrow$ Paley conference matrices. (Size: 10009×10009)
- For each $p \in \{0.1, 0.25, 0.4\}$ we draw a subset of indices $\mathcal{I}$ from $\{1, \cdots, 10009\}$ including each index independently with probability p.
- Take the principal submatrix using $\mathcal{I}$ as the subindices.
- The histograms of the eigenvalues of the submatrices appear to form a Kesten–McKay distribution with parameter $v = 1/p$!



Paley ETFs seem to exhibit a “probabilistic RIP”. Is this coincidence?

Paley ETFs **really** do exhibit RIP-like behavior.

Theorem (M., Mixon, Parshall '20)

Fix $p \in (0, \frac{1}{2})$, and take any sequence $\{a_n : n \in \mathbb{R}\}$ for which

1. *there exists $\lambda > 1$ such that $a_{n+1} \geq \lambda a_n$ for every n, and*
2. *there exists a symmetric conference matrix $\mathbf{S}_n$ of size $a_n \times a_n$ for each n.*

For each n randomly draw $\mathcal{I}_n$ from $\{1, \cdots, a_n\}$ including each index independently with probability p, and define the random matrix $\mathbf{X}_n$ to be the principal submatrix of $\mathbf{S}_n$ with rows and columns indexed by $\mathcal{I}_n$. Then, the emperical spectral distribution of $\frac{1}{p\sqrt{n}}\mathbf{X}_n$ converges almost surely to the Kesten–McKay distribution with parameter $v = 1/p$.

Paley ETFs **really** do exhibit RIP-like behavior.

Theorem (M., Mixon, Parshall '20)

Fix $p \in (0, \frac{1}{2})$, *and take any sequence* $\{a_n : n \in \mathbb{R}\}$ *for which*

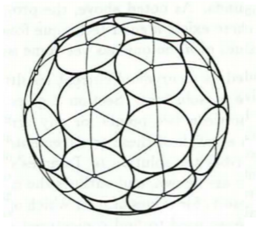
1. *there exists* $\lambda > 1$ *such that* $a_{n+1} \geq \lambda a_n$ *for every* n, *and*
2. *there exists a symmetric conference matrix* $\mathbf{S}_n$ *of size* $a_n \times a_n$ *for each* n.

For each n *randomly draw* $\mathcal{I}_n$ *from* $\{1, \cdots, a_n\}$ *including each index independently with probability* p, *and define the random matrix* $\mathbf{X}_n$ *to be the principal submatrix of* $\mathbf{S}_n$ *with rows and columns indexed by* $\mathcal{I}_n$. *Then, the emperical spectral distribution of* $\frac{1}{p\sqrt{n}}\mathbf{X}_n$ *converges almost surely to the Kesten–McKay distribution with parameter* $v = 1/p$.

The theorem is super technical, but it confirms the picture on the previous slide is no accident!

Table of Contents

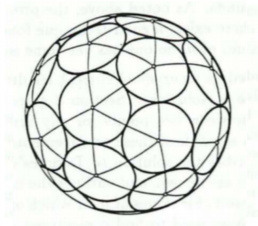
Packing problems are everywhere![2]



1. Spreading pollen pores
2. British soldiers stacking cannonballs
3. Quantum measurement: ETFs of d^2 vectors in $\mathbb{C}^d$.
4. Coding theory
5. Oncology

[2] Tarnai 1983 – 24 spherical caps solution

Packing problems are everywhere![2]



1. Spreading pollen pores
2. British soldiers stacking cannonballs
3. Quantum measurement: ETFs of d^2 vectors in $\mathbb{C}^d$.
4. Coding theory
5. Oncology

Optimal packing problems arise naturally in many applications. How does frame theory help us analyze them?

[2] Tarnai 1983 – 24 spherical caps solution

ETFs are optimal packings.

Let $X = \{v_1, \cdots, v_n\}$ be a set of $n \geq d$ unit vectors in $\mathbb{R}^d$. The coherence of X is computed by

$$\mu(X) := \max_{i \neq j} |v_i \cdot v_j|.$$

Maximum inner product $\leftrightarrow$ Minimum interior angle

Theorem (Welch–Rankin Bound)

Let $X = \{v_1, \cdots, v_n\}$ be a set of $n \geq d$ unit vectors. Then,

$$\mu(X) \geq \sqrt{\frac{n-d}{d(n-1)}}.$$

Equality is achieved if and only if X is an **equiangular tight frame**.

ETFs are optimal packings.

Let $X = \{v_1, \cdots, v_n\}$ be a set of $n \geq d$ unit vectors in $\mathbb{R}^d$. The coherence of X is computed by

$$\mu(X) := \max_{i \neq j} |v_i \cdot v_j|.$$

Maximum inner product $\leftrightarrow$ Minimum interior angle

Theorem (Welch–Rankin Bound)

Let $X = \{v_1, \cdots, v_n\}$ be a set of $n \geq d$ unit vectors. Then,

$$\mu(X) \geq \sqrt{\frac{n-d}{d(n-1)}}.$$

Equality is achieved if and only if X is an **equiangular tight frame**.

ETFs are optimal packings when they exist. When they don't, can we do better than this bound?

We can do better when there are no ETFs...sometimes.

Important labels:
Black: Known Packings, Blue: Welch–Rankin, Green: Bukh–Cox

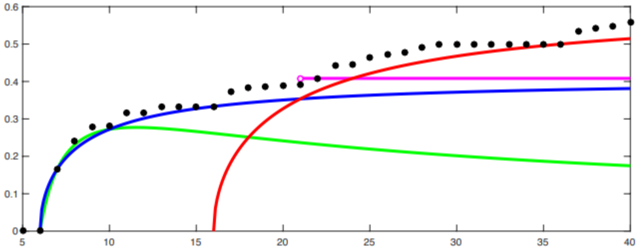


Figure: Coherence of best known packings and some bounds in $\mathbb{R}^6$.

We can do better when there are no ETFs...sometimes.

Important labels:
Black: Known Packings, Blue: Welch–Rankin, Green: Bukh–Cox

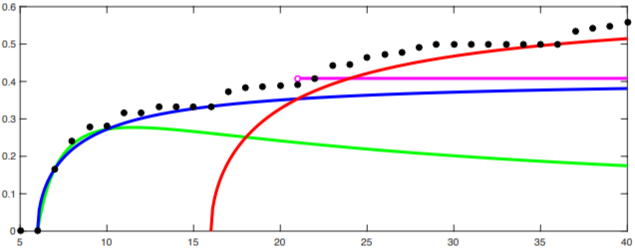


Figure: Coherence of best known packings and some bounds in $\mathbb{R}^6$.

Improvement in the $n \geq 21$ case is expected. Why with the $n \leq 10$ range?

The improvement for small n might be improvable!

Theorem (Bukh–Cox '18)

Let $X = \{v_1, \cdots, v_n\}$ be a set of $n \geq d$ unit vectors in $\mathbb{R}^d$. Then,

$$\mu(X) \geq \frac{(n-d)(n-d+1)}{2n + (n^2 - nd - n)\sqrt{2+n-d} - (n-d)(n-d+1)}.$$

- Bukh, Cox '18: Original combinatorial proof, very long and technical
- M., Mixon, Parshall '19: Shorter proof using a classic linear programming method.
- **You (?)**: These methods exist for **subspace** packings. Extensions to these settings?

The improvement for small n might be improvable!

Theorem (Bukh–Cox '18)

Let $X = \{v_1, \cdots, v_n\}$ be a set of $n \geq d$ unit vectors in $\mathbb{R}^d$. Then,

$$\mu(X) \geq \frac{(n-d)(n-d+1)}{2n + (n^2 - nd - n)\sqrt{2+n-d} - (n-d)(n-d+1)}.$$

- Bukh, Cox '18: Original combinatorial proof, very long and technical
- M., Mixon, Parshall '19: Shorter proof using a classic linear programming method.
- **You (?)**: These methods exist for **subspace** packings. Extensions to these settings?

It might be possible to extend Bukh and Cox's work to other packing problems. Where else does frame theory arise in packing?

ETFs are optimal quantum measurement systems.

- Quantum State Tomography: optimal measurement system $\rightarrow$ ETF of d^2 in $\mathbb{C}^d$.
- These are called SIC-POVMs or simply, SICs.
- **Zauner's Conjecture:** There exists a SIC in every dimension.
- Numerical constructions exist for every dimension up to $d = 151$.
- Exact constructions in a handful of dimensions, e.g. $d = 2, 24, 28, 48, 124$.

ETFs are optimal quantum measurement systems.

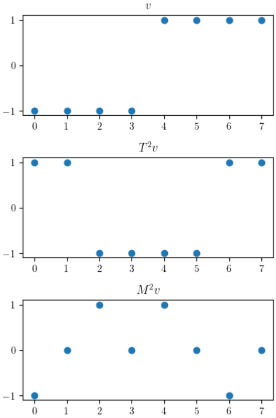
- Quantum State Tomography: optimal measurement system $\rightarrow$ ETF of d^2 in $\mathbb{C}^d$.
- These are called SIC-POVMs or simply, SICs.
- **Zauner's Conjecture:** There exists a SIC in every dimension.
- Numerical constructions exist for every dimension up to $d = 151$.
- Exact constructions in a handful of dimensions, e.g. $d = 2, 24, 28, 48, 124$.

SICs are optimal quantum measurements but we can't prove they always exist. What do we do?

With the right vectors, groups can create SICs.

How most known SICs are made:

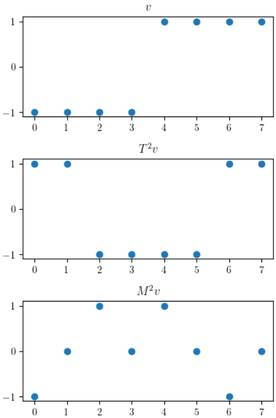
- Take a special unit vector $v \in \mathbb{C}^d$.
- Act on it with the Heisenberg group.
  - All time and frequency shifts.
- Exceptions: Wigner SICs, Hoggar lines



With the right vectors, groups can create SICs.

How most known SICs are made:

- Take a special unit vector $v \in \mathbb{C}^d$.
- Act on it with the Heisenberg group.
  - All time and frequency shifts.
- Exceptions: Wigner SICs, Hoggar lines



Most SICs are generated by groups, but finding the special vectors is hard.
How do people find them?

The ways of finding SICs varies greatly.

What are people trying?

- Computer simulations: numerical $\rightarrow$ exact
- Number Theory: Connection to Stark conjectures.
- Combinatorial designs
- (M. + Mixon): Relax to a bigger space $\rightarrow$ leads to **algebraic variety** that seems to be 1-dimensional...

The ways of finding SICs varies greatly.

What are people trying?

- Computer simulations: numerical $\rightarrow$ exact
- Number Theory: Connection to Stark conjectures.
- Combinatorial designs
- (M. + Mixon): Relax to a bigger space $\rightarrow$ leads to **algebraic variety** that seems to be 1-dimensional...

People are pursuing SICs in many different ways. Our approach has us stuck on algebraic geometry (please help!).

We can work together!

- Computer simulations to explore random submatrices of other ETFs.
- Make a Bukh–Cox equivalent for subspace packings.
- Explore the biangular curve further.
- Heisenberg group is the core of time-frequency analysis. Study generalizions of frames generalized by this group in discrete and continuous settings.
- A certain class of error-correcting codes can also be described by an algebraic variety or as circulant Hadamard matrices. Explore this further with computer simluations.

Thanks for listening! Questions?