Vector Subspace Structure and FRI

Guillermo Angeris Alex Evans

Session 3, SPLA Study Club

Outline

Quick recap

- We've studied all main tools necessary
- And most of the major checks
- (Either via homework or lecture!)
- Now, we're going to do something real with them!

Main idea

- The main point will be to try and understand FRI
- To do this, we need some FRI-specific set up
- Namely: decomposing a vector space recursively

Main idea

- The main point will be to try and understand FRI
- To do this, we need some FRI-specific set up
- Namely: decomposing a vector space recursively
- Similar to previous: start with exact case
- Move to inexact case (which is ‘harder’)

Outline

Subspaces and decomposition

- We start back with subspaces
- Let $V \subseteq \mathbf{F}^n$ be a subspace (along with $V', V'' \subseteq \mathbf{F}^n$)
- We say V *decomposes* into V' and V'' if, for each $x \in V$,

$$x = y + z,$$

  for some $y \in V'$ and $z \in V''$
- We write this as

$$V = V' + V''$$

An example

- A simple example: let $V = \mathbf{F}^2$, then

$$V = \{(\alpha, 0) \mid \alpha \in \mathbf{F}\} + \{(0, \beta) \mid \beta \in \mathbf{F}\}$$

A (more complicated) example

- A second example is: let V be the Reed–Solomon codewords

$$V = \{(f(\alpha_1), \ldots, f(\alpha_m)) \mid f \text{ has degree} \leq n - 1\}$$

A (more complicated) example

- A second example is: let V be the Reed–Solomon codewords

$$V = \{(f(\alpha_1), \ldots, f(\alpha_m)) \mid f \text{ has degree } \leq n-1\}$$

- If we define the following subspaces (check this!)

$$V' = \{(f(\alpha_1), \ldots, f(\alpha_m)) \mid f \text{ has even powers, deg. } \leq n-1\}$$
$$V'' = \{(f(\alpha_1), \ldots, f(\alpha_m)) \mid f \text{ has odd powers, deg. } \leq n-1\}$$

- Then

$$V = V' + V''$$

Not a difficult observation

- Note that, if n is even

$$f(\alpha) = x_1 + x_2\alpha + x_3\alpha^2 + \cdots + x_n\alpha^{n-1}$$

Not a difficult observation

- Note that, if n is even

$$f(\alpha) = x_1 + x_2\alpha + x_3\alpha^2 + \cdots + x_n\alpha^{n-1}$$

- Same as saying:

$$f(\alpha) = \underbrace{x_1 + x_3\alpha^2 + \dots}_{\text{even powers}} + \underbrace{x_2\alpha + x_4\alpha^3 + \dots}_{\text{odd powers}}$$

Recursive decomposition

- Given a vector space $V \subseteq \mathbf{F}^n$ and a matrix $T \in \mathbf{F}^{m \times n}$, then

$$TV = \{ Tx \mid x \in V \}$$

  is also a vector space

Recursive decomposition

- Given a vector space $V \subseteq \mathbf{F}^n$ and a matrix $T \in \mathbf{F}^{m \times n}$, then

$$TV = \{\, Tx \mid x \in V \,\}$$

  is also a vector space

- (Homework problem!)

Recursive decomposition

- Given a vector space $V \subseteq \mathbf{F}^n$ and a matrix $T \in \mathbf{F}^{m \times n}$, then

$$TV = \{ Tx \mid x \in V \}$$

is also a vector space

- (Homework problem!)

- We say this subspace *recursively decomposes* if

$$V = T_1 V' + T_2 V',$$

for $V' \subseteq \mathbf{F}^k$ and $T_1, T_2 \in \mathbf{F}^{n \times k}$

Simple example

- Same example as before! If $V = \mathbf{F}^2$, then

$$V = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \mathbf{F} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{F}$$

Simple example

- Same example as before! If $V = \mathbf{F}^2$, then

$$V = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \mathbf{F} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{F}$$

- Or, more generally, if $V = \mathbf{F}^{2n}$, then

$$V = \begin{bmatrix} I \\ 0 \end{bmatrix} \mathbf{F}^n + \begin{bmatrix} 0 \\ I \end{bmatrix} \mathbf{F}^n$$

(See homework!)

Reed–Solomon example

- From before, let V be the Reed–Solomon codewords, n even

$$V = \{(f(\alpha_1), \ldots, f(\alpha_m)) \mid f \text{ has degree } \leq n-1\}$$

- And define (as before)

$$V' = \{(f(\alpha_1), \ldots, f(\alpha_m)) \mid f \text{ has even powers, deg. } \leq n-1\}$$

Reed–Solomon example

- From before, let V be the Reed–Solomon codewords, n even

$$V = \{(f(\alpha_1), \dots, f(\alpha_m)) \mid f \text{ has degree} \leq n-1\}$$

- And define (as before)

$$V' = \{(f(\alpha_1), \dots, f(\alpha_m)) \mid f \text{ has even powers, deg.} \leq n-1\}$$

- Then, V can be written $V = V' + DV'$, where

$$D = \begin{bmatrix} \alpha_1 & 0 & \dots & 0 \\ 0 & \alpha_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \alpha_m \end{bmatrix}$$

(See homework!)

Outline

How do we use this?

- Clearly, these spaces have a lot of structure
- How can we use it?

How do we use this?

- Clearly, these spaces have a lot of structure
- How can we use it?
- Let's design a (simple?) protocol!

A simple check

- Want to check if $x \in V$ when $V = T_1 V' + T_2 V'$

A simple check

- Want to check if $x \in V$ when $V = T_1 V' + T_2 V'$
- Suffices to check that
  1. The equality $x = T_1 y + T_2 z$ holds
  2. Subspace inclusion is satisfied: $y, z \in V'$

A simple check

- Want to check if $x \in V$ when $V = T_1 V' + T_2 V'$
- Suffices to check that
  1. The equality $x = T_1 y + T_2 z$ holds
  2. Subspace inclusion is satisfied: $y, z \in V'$
- Savings if checking that $y, z \in V'$ takes at least $\sim n^2$ time

A simple check

- Want to check if $x \in V$ when $V = T_1 V' + T_2 V'$
- Suffices to check that
  1. The equality $x = T_1 y + T_2 z$ holds
  2. Subspace inclusion is satisfied: $y, z \in V'$
- Savings if checking that $y, z \in V'$ takes at least $\sim n^2$ time
- If V' decomposes further, then more savings!

Where is the randomness?

- From before, let $G \in \mathbf{F}^{m \times 2}$ with distance d
- Remembering the vector subspace check from session 2:

$$G_{r1}y + G_{r2}z \in V' \underset{p}{\implies} y, z \in V',$$

  where $p \leq 1 - d/m$
- With this additional step, we have a protocol with error $\leq p$
  1. Check equality $x = T_1 y + T_2 z$ holds
  2. Draw random r from $1, \ldots, m$
  3. Verify that $G_{r1}y + G_{r2}z \in V'$

Where is the randomness?

- Protocol from before:
  1. Check equality $x = T_1 y + T_2 z$ holds
  2. Draw random r from $1, \ldots, m$
  3. Verify that $G_{r1} y + G_{r2} z \in V'$
- Why does this imply that $x \in V$ with error probability $\leq p$?

Outline

Exact checks

- Note that the previous ‘protocol’ requires exact equality
- (And exact inclusion!)

Exact checks

- Note that the previous 'protocol' requires exact equality
- (And exact inclusion!)
- Hard to achieve unless we are in the coding model
- For the same reason as the exact zero check

Distance check

- On the other hand, say we're in the direct access model
- How can we relax these conditions?

Distance check

- On the other hand, say we're in the direct access model
- How can we relax these conditions?
- First, relax exact equality to ‘close enough’

Distance check

- On the other hand, say we're in the direct access model
- How can we relax these conditions?
- First, relax exact equality to 'close enough'
- Second, relax exact inclusion to 'close enough' :)

Distance check

- On the other hand, say we're in the direct access model
- How can we relax these conditions?
- First, relax exact equality to 'close enough'
- Second, relax exact inclusion to 'close enough' :)
- What does this mean?

Distance to a subspace

- As usual, distance between two vectors $x, y \in \mathbf{F}^n$ is defined

$$\|x - y\|_0$$

- (This is easy-ish via sparsity check! See previous homework)

Distance to a subspace

- As usual, distance between two vectors $x, y \in \mathbf{F}^n$ is defined

$$\|x - y\|_0$$

- (This is easy-ish via sparsity check! See previous homework)
- Given a subspace $V \subseteq \mathbf{F}^n$, the *distance* of x to V is written

$$\|x - V\|_0 = \min_{y \in V} \|x - y\|_0$$

Matrix distance

- Given a matrix X, we say

$$\Delta(X, V) \leq q$$

if there is a matrix Y, with columns in V, such that $X - Y$ has at most q nonzero rows (mouthful!)

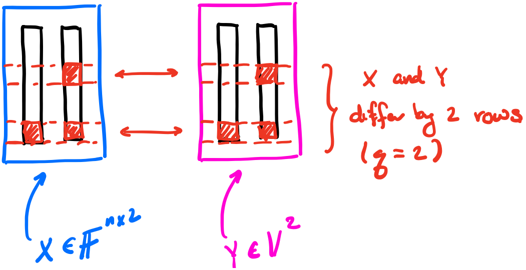
Matrix distance

- Given a matrix X, we say

$$\Delta(X, V) \leq q$$

if there is a matrix Y, with columns in V, such that $X - Y$ has at most q nonzero rows (mouthful!)

- The picture is



Revised protocol

- Start with a simple revision: let q be an 'acceptable distance'
  1. Check that $\|x - (T_1 y + T_2 z)\|_0 \leq q$
  2. Draw random r from $1, \ldots, m$
  3. Check that $\|G_{r1} y + G_{r2} z - V'\|_0 \leq q$
- Does this guarantee that $\|x - V\| \leq Cq$ for some C?

Revised protocol

- Start with a simple revision: let q be an 'acceptable distance'
  1. Check that $\|x - (T_1 y + T_2 z)\|_0 \leq q$
  2. Draw random r from $1, \ldots, m$
  3. Check that $\|G_{r1} y + G_{r2} z - V'\|_0 \leq q$
- Does this guarantee that $\|x - V\| \leq Cq$ for some C?
- Remember, T_1 and T_2 can be very badly structured!

Structure on T_i

- Unfortunately, sparsity depends on structure

Structure on T_i

- Unfortunately, sparsity depends on structure
- We need one more definition: T_1 and T_2 are *basis aligned*
- Whenever $[y\ z]$ has $\leq q$ nonzero rows, then

$$T_1 y + T_2 z$$

  has $\leq q'$ nonzero entries

Structure on T_i

- Unfortunately, sparsity depends on structure
- We need one more definition: T_1 and T_2 are *basis aligned*
- Whenever $[y \; z]$ has $\leq q$ nonzero rows, then

$$T_1 y + T_2 z$$

  has $\leq q'$ nonzero entries
- Example: $q = q'$ if T_1 and T_2 are only diagonal (see homework!)

High level of basis alignment

- Essentially, basis alignment means if y and z are q-close to V'
- Then, we have that $T_1 y + T_2 z$ must be q'-close to V
- And this is the last step we need!

(One step of) the FRI protocol

- Let V be the set of Reed–Solomon codes of degree $2k$
- And V' is the same set with degree k (instead of $2k$)
- Note that protocol is
  1. Check that $\|x - (T_1 y + T_2 z)\|_0 \leq q$
  2. Draw random r from $1, \ldots, m$
  3. Check that $\|G_{r1} y + G_{r2} z - V'\|_0 \leq q$

(One step of) the FRI protocol

- Let V be the set of Reed–Solomon codes of degree $2k$
- And V' is the same set with degree k (instead of $2k$)
- Note that protocol is
  1. Check that $\|x - (T_1 y + T_2 z)\|_0 \leq q$
  2. Draw random r from $1, \ldots, m$
  3. Check that $\|G_{r1} y + G_{r2} z - V'\|_0 \leq q$
- First is a sparse check, third is a subspace distance check
- Implies that $\|x - V\|_0 \leq 2q$ (with some probability)

Recursive application

- Since V' is also an RS code, then it too decomposes!
- Instead of checking the last step

$$\|G_{r1}y + G_{r2}z - V'\|_0 \leq q$$

- Run the protocol again, over new vector

$$w = G_{r1}y + G_{r2}z$$

Recursive application

- Since V' is also an RS code, then it too decomposes!
- Instead of checking the last step

$$\|G_{r1}y + G_{r2}z - V'\|_0 \leq q$$

- Run the protocol again, over new vector

$$w = G_{r1}y + G_{r2}z$$

- Terminate at some later point by just checking inclusion

Result

- If we repeat this k times, we get that

$$\|x - V\| \leq 2q$$

  with high probability

- The details are a bit messy (though ultimately easy)
- See paper §4.2

High level result

- We started with checking x close to V
- Reduced it to getting y and z which have two properties

High level result

- We started with checking x close to V
- Reduced it to getting y and z which have two properties
- First, $T_1 y + T_2 z$ is close to x

High level result

- We started with checking x close to V
- Reduced it to getting y and z which have two properties
- First, $T_1 y + T_2 z$ is close to x
- Second y and z are each close to V'
- 'Reduce' this further to checking $G_{r1} y + G_{r2} z$ is close to V'

High level result

- We started with checking x close to V
- Reduced it to getting y and z which have two properties
- First, $T_1 y + T_2 z$ is close to x
- Second y and z are each close to V'
- 'Reduce' this further to checking $G_{r1} y + G_{r2} z$ is close to V'
- And then iterate!

High level discussion

- Note that polynomials aren't really needed
- But they make a *great* practical case
- (Possible that there may be structured codes that are useful)

Outline

La fin

- With all of that; we have ‘finished’ the course
- Many tools, one example application, but likely many more
- Encourage you to go and try to write others using this!
- Please PR for homework and slide feedback if needed :)

La fin

- With all of that; we have ‘finished’ the course
- Many tools, one example application, but likely many more
- Encourage you to go and try to write others using this!
- Please PR for homework and slide feedback if needed :)
- And thanks for attending!