(Nearly) Sample Optimal Sparse Fourier Transform

Piotr Indyk[1] **Michael Kapralov** [1] Eric Price[2]

[1]MIT

[2]MIT→ IBM Almaden → UT Austin

SODA'14

Discrete Fourier Transform

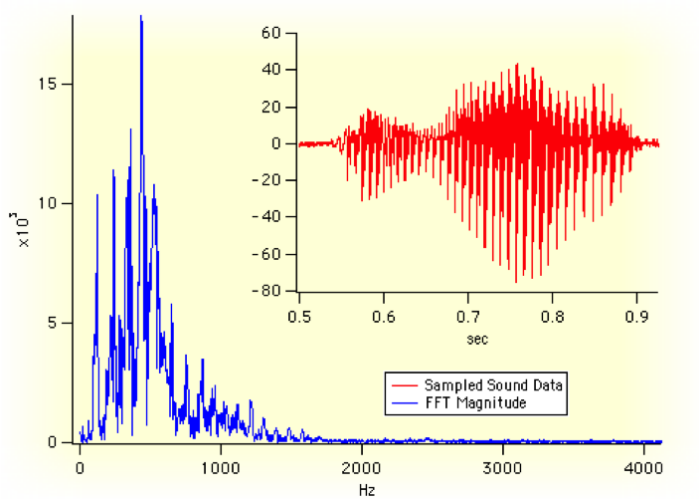
Given $x \in \mathbb{C}^n$, compute

$$\widehat{x}_i = \sum_{j \in [n]} x_j \omega^{ij},$$

where ω is the n-th root of unity.

Fundamental tool:

- Compression (image, audio, video)
- Signal processing
- Data analysis
- Medical imaging (MRI, NMR)



Sparse Fourier Transform

The fast algorithm for DFT is FFT, runs in $O(n \log n)$ time

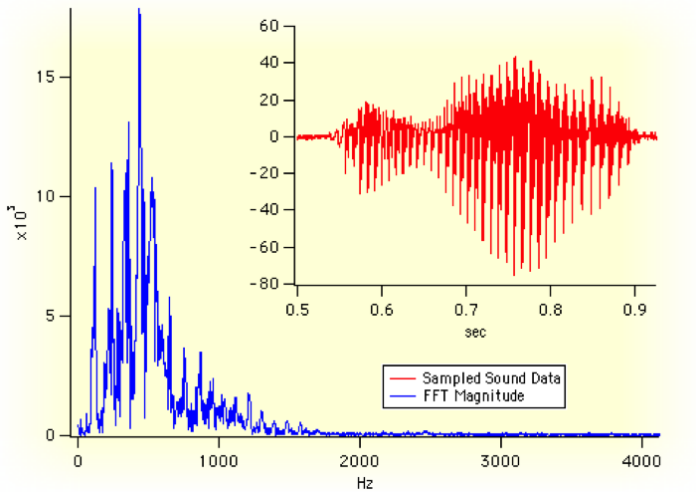
- improving on FFT in full generality a major open problem

Sparse Fourier Transform

The fast algorithm for DFT is FFT, runs in $O(n \log n)$ time

- improving on FFT in full generality a major open problem

Most interesting signals are sparse (have few nonzero entries) or approximately sparse in the Fourier domain.



k-sparse=at most k non-zeros

Hassanieh-Indyk-Katabi-Price'12 compute approximate sparse FFT in $O(k \log n \log(n/k))$ time

Sample complexity

Sample complexity=number of samples accessed in time domain.
In some applications at least as important as runtime

Magnetic Resonance Imaging

Sample complexity

Sample complexity=number of samples accessed in time domain.

In some applications at least as important as runtime

Magnetic Resonance Imaging

Given access to $x \in \mathbb{C}^n$, find $\widehat{y}$ such that

$$||\widehat{x} - \widehat{y}||^2 \leq C \cdot \min_{k-\text{sparse } \widehat{z}} ||\widehat{x} - \widehat{z}||^2$$

Optimal sample complexity?

...and small runtime?

Uniform bounds (for all):

Candes-Tao'06
Rudelson-Vershynin'08
Cheraghchi-Guruswami-Velingker'12

Deterministic, $\Omega(n)$ runtime

$O(k \log^3 k \log n)$

Non-uniform bounds (for each):

Goldreich-Levin'89
Mansour'92
Gilbert-Guha-Indyk-Muthukrishnan-Strauss'02
Gilbert-Muthukrishnan-Strauss'05
Hassanieh-Indyk-Katabi-Price'12a
Hassanieh-Indyk-Katabi-Price'12b

Randomized, $O(k \cdot \text{poly}(\log n))$ runtime

$O(k \log n \log(n/k))$

Lower bound: $\Omega(k \log n / \log\log n)$ even for adaptive algorithms Hassanieh-Indyk-Katabi-Price'12

Uniform bounds (for all):

Candes-Tao'06
Rudelson-Vershynin'08
Cheraghchi-Guruswami-Velingker'12

Deterministic, $\Omega(n)$ runtime

$O(k \log^3 k \log n)$

Non-uniform bounds (for each):

Goldreich-Levin'89
Mansour'92
Gilbert-Guha-Indyk-Muthukrishnan-Strauss'02
Gilbert-Muthukrishnan-Strauss'05
Hassanieh-Indyk-Katabi-Price'12a
Hassanieh-Indyk-Katabi-Price'12b

Randomized, $O(k \cdot \mathrm{poly}(\log n))$ runtime

$O(k \log n \log(n/k))$

Lower bound: $\Omega(k \log n / \log\log n)$ even for adaptive algorithms Hassanieh-Indyk-Katabi-Price'12

Theorem

There exists an algorithm for ℓ_2/ℓ_2 sparse recovery from Fourier measurements using $O^*(k \log n)$ samples and $O^*(k \log^2 n)$ runtime.

$O^*()$ hides $(\log\log n)^{O(1)}$ factors.

Optimal up to $O((\log\log n)^C)$ factor

ℓ_2/ℓ_2 sparse recovery guarantees:

$$||\widehat{x} - \widehat{y}||^2 \leq C \cdot \min_{k-\text{sparse } \widehat{z}} ||\widehat{x} - \widehat{z}||^2$$

ℓ_2/ℓ_2 sparse recovery guarantees:

$$||\widehat{x} - \widehat{y}||^2 \leq C \cdot \mathrm{Err}_k^2(\widehat{x})$$

$|\widehat{x}_1| \geq \ldots \geq |\widehat{x}_k| \geq |\widehat{x}_{k+1}| \geq |\widehat{x}_{k+2}| \geq \cdots$

$$\mathrm{Err}_k^2(\widehat{x}) = \sum_{j=k+1}^{n} |\widehat{x}_j|^2$$

Residual error bounded by noise energy $\mathrm{Err}_k^2(\widehat{x})$

ℓ_2/ℓ_2 sparse recovery guarantees:

Signal to noise ratio $R = ||\widehat{x} - \widehat{y}||^2 / \mathrm{Err}_k^2(\widehat{x}) \leq C$

$|\widehat{x}_1| \geq \ldots \geq |\widehat{x}_k| \geq |\widehat{x}_{k+1}| \geq |\widehat{x}_{k+2}| \geq \cdots$

$\mathrm{Err}_k^2(\widehat{x}) = \sum_{j=k+1}^{n} |\widehat{x}_j|^2$

Residual error bounded by noise energy $\mathrm{Err}_k^2(\widehat{x})$

ℓ_2/ℓ_2 sparse recovery guarantees:

$$\text{Signal to noise ratio } R = ||\widehat{x} - \widehat{y}||^2 / \mathrm{Err}_k^2(\widehat{x}) \leq C$$

$|\widehat{x}_1| \geq \ldots \geq |\widehat{x}_k| \geq |\widehat{x}_{k+1}| \geq |\widehat{x}_{k+2}| \geq \cdots$

$$\mathrm{Err}_k^2(\widehat{x}) = \sum_{j=k+1}^{n} |\widehat{x}_j|^2$$

Residual error bounded by noise energy $\mathrm{Err}_k^2(\widehat{x})$

Sufficient to ensure that most elements are below average noise level:

$$|\widehat{x}_i - \widehat{y}_i|^2 \leq c \cdot \mathrm{Err}_k^2(\widehat{x})/k =: \mu^2$$

Iterative recovery

Many algorithms use the iterative recovery scheme:

Input: $x \in \mathbb{C}^n$
$\widehat{y}_0 \leftarrow 0$
For $t = 1$ to L

- $\widehat{z} \leftarrow \text{REFINEMENT}(x, \widehat{y}_{t-1})$ $\quad \triangleright$ Takes random samples of $x - y$
- Update $\widehat{y}_t \leftarrow \widehat{y}_{t-1} + \widehat{z}$

REFINEMENT$(x, \widehat{y})$

return dominant Fourier coefficients $\widehat{z}$ of $x - y$ (approximately)

dominant coefficients$\approx |\widehat{x}_i - \widehat{y}_i|^2 \geq \mu^2$(above average noise level)

REFINEMENT$(x, \widehat{y})$

return dominant Fourier coefficients $\widehat{z}$ of $x - y$ (approximately)

dominant coefficients$\approx |\widehat{x}_i - \widehat{y}_i|^2 \geq \mu^2$(above average noise level)

Main questions:

- How many samples per SNR reduction step?
- How many iterations?

REFINEMENT$(x, \widehat{y})$

return dominant Fourier coefficients $\widehat{z}$ of $x - y$ (approximately)

dominant coefficients$\approx |\widehat{x}_i - \widehat{y}_i|^2 \geq \mu^2$(above average noise level)

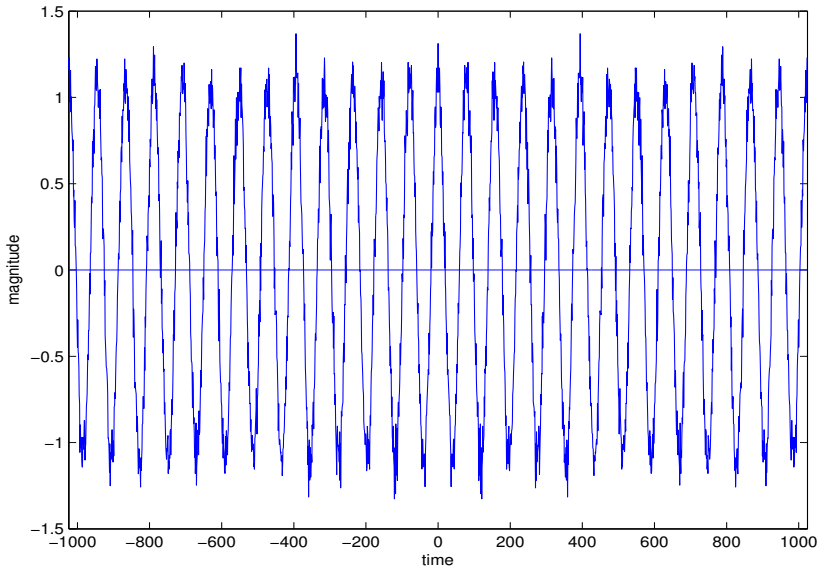
Main questions:

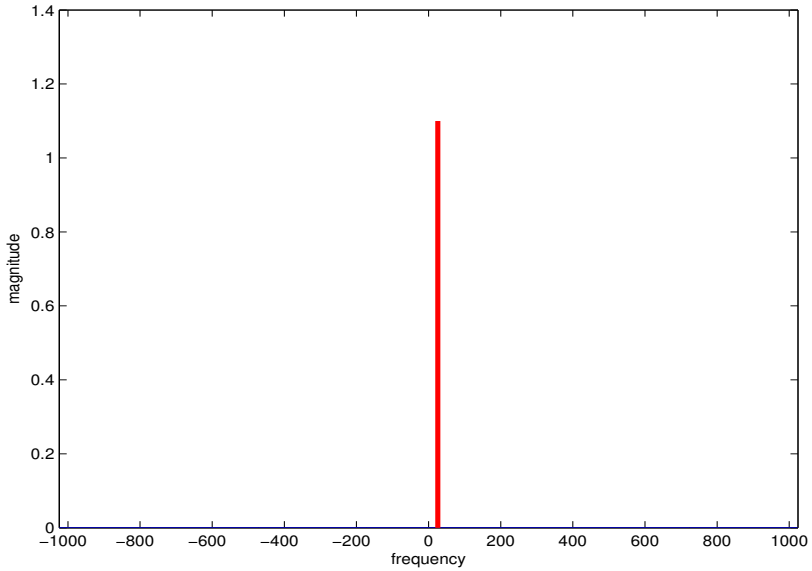
- How many samples per SNR reduction step?
- How many iterations?

Summary of techniques from

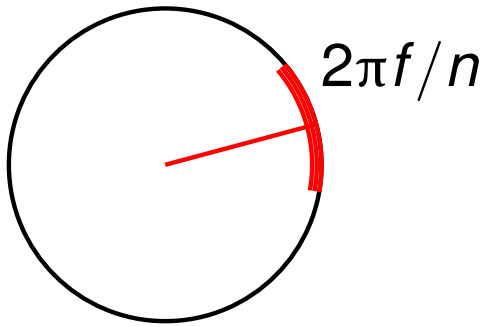
Gilbert-Guha-Indyk-Muthukrishnan-Strauss'02, Akavia-Goldwasser-Safra'03, Gilbert-Muthukrishnan-Strauss'05, Iwen'10, Akavia'10, Hassanieh-Indyk-Katabi-Price'12a, Hassanieh-Indyk-Katabi-Price'12b

1-sparse recovery from Fourier measurements





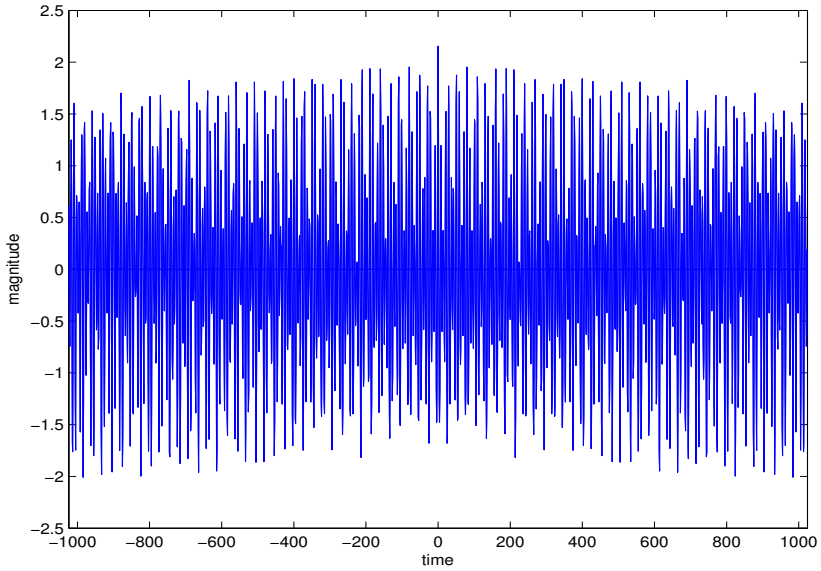
$$x_a = \omega^{a \cdot f} + \text{noise}$$

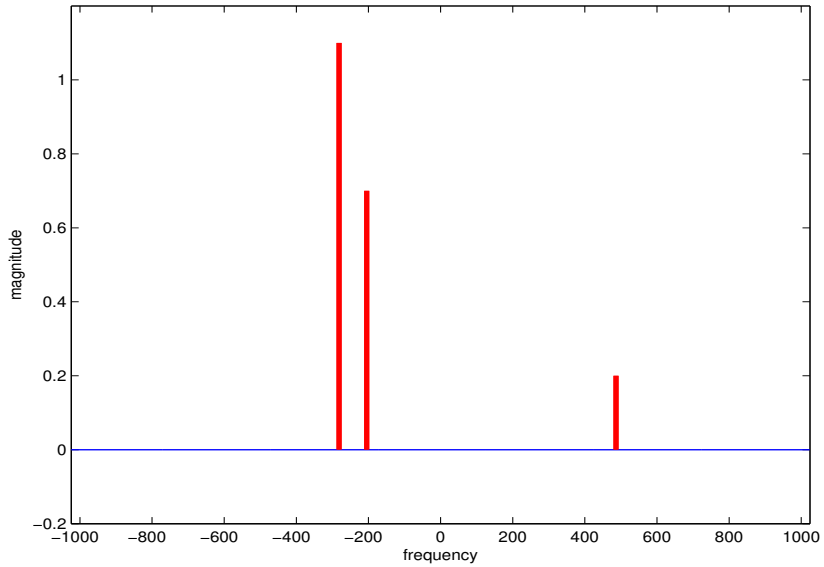


$O(\log_{SNR} n)$ measurements

Reducing k-sparse recovery to 1-sparse recovery

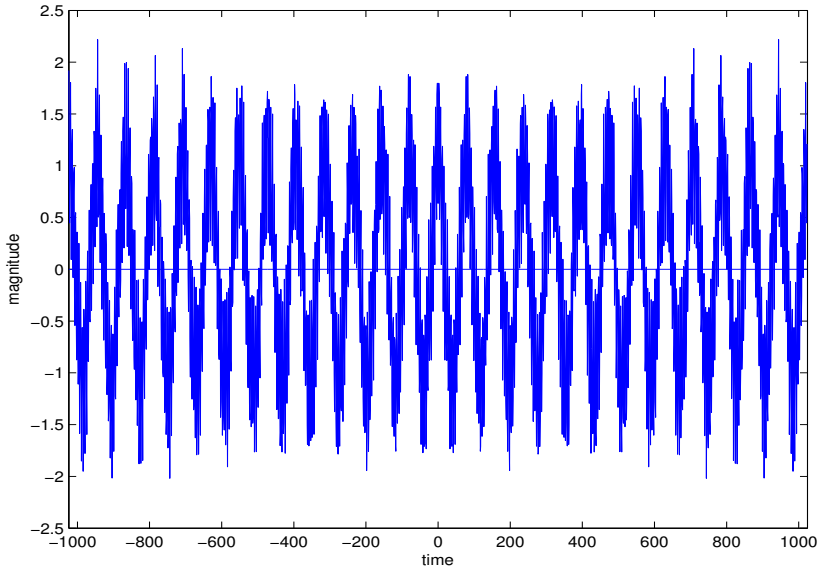
Permute with a random linear transformation and phase shift

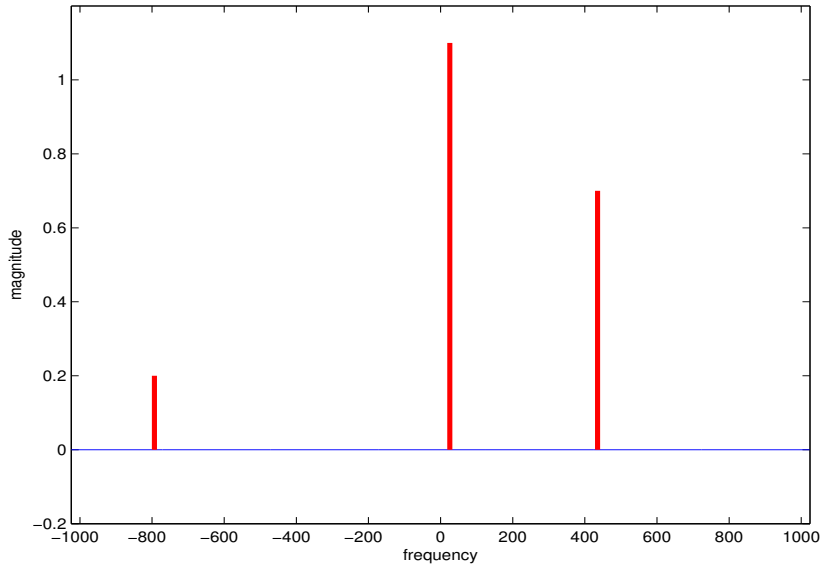




Reducing k-sparse recovery to 1-sparse recovery

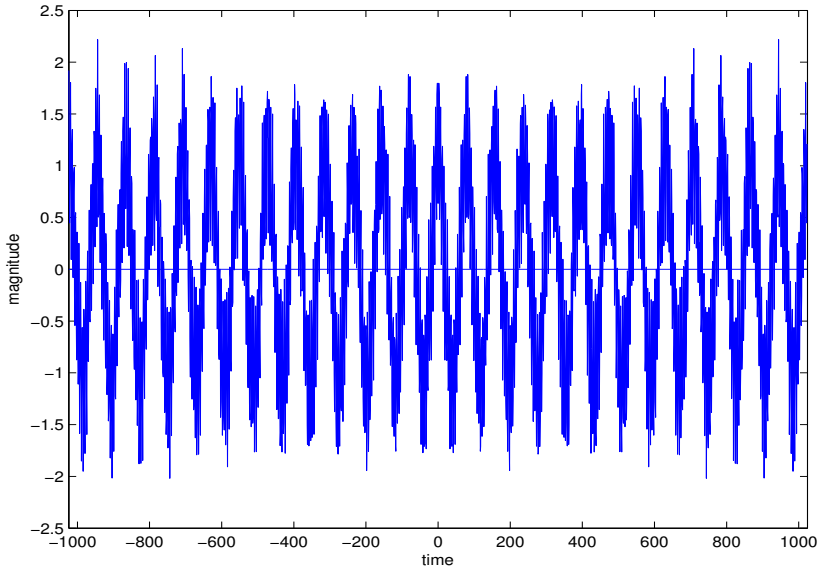
Permute with a random linear transformation and phase shift

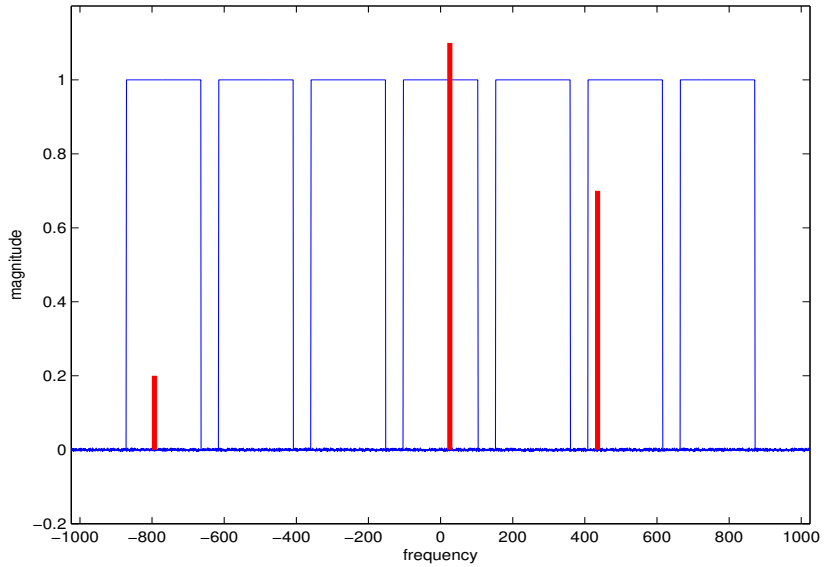




Reducing k-sparse recovery to 1-sparse recovery

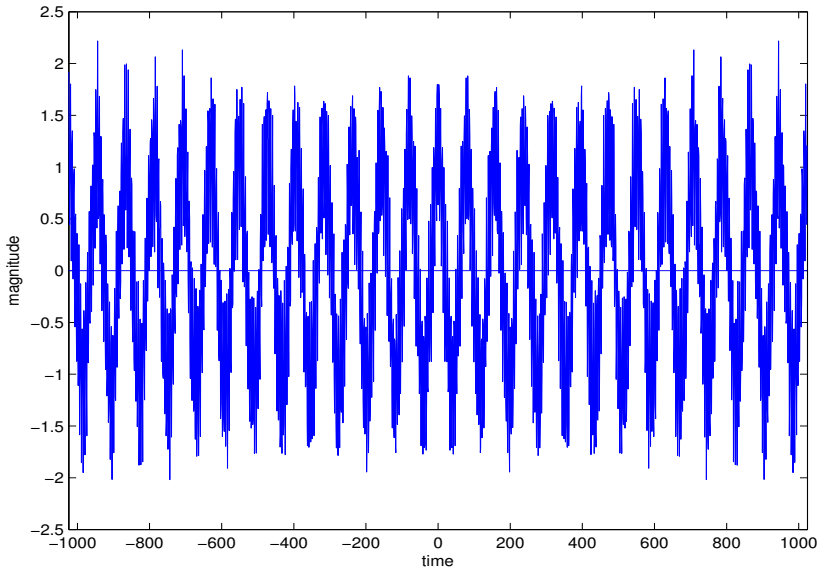
Permute with a random linear transformation and phase shift

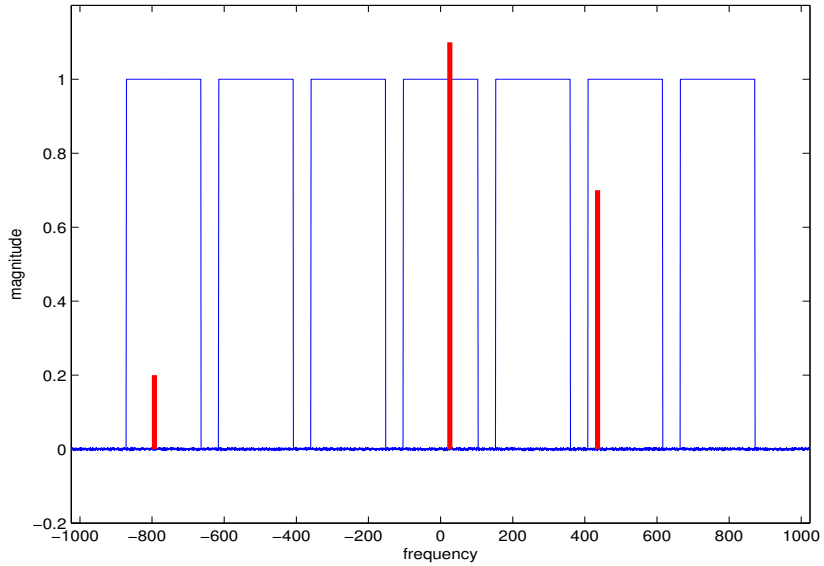




Reducing k-sparse recovery to 1-sparse recovery

Partition frequency space into $B = k/\alpha$ buckets for constant $\alpha \in (0,1)$





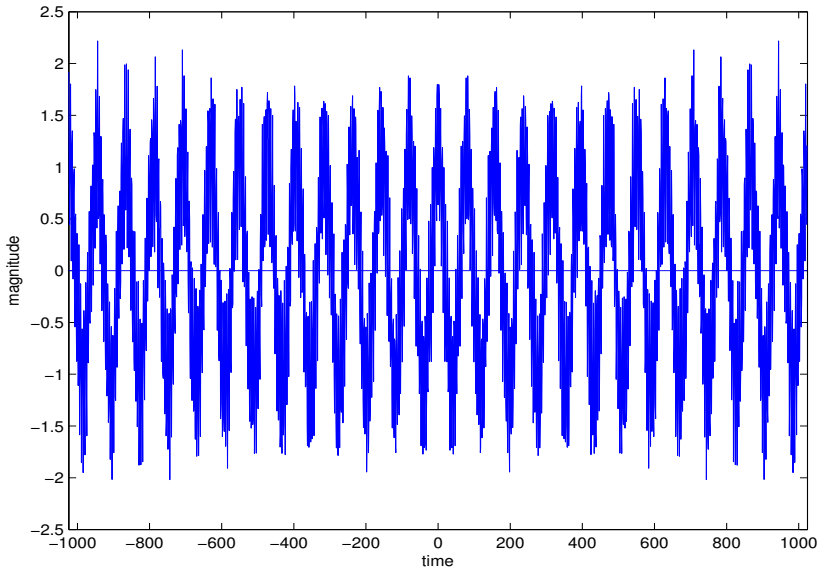
Choose a filter $G, \hat{G}$ such that

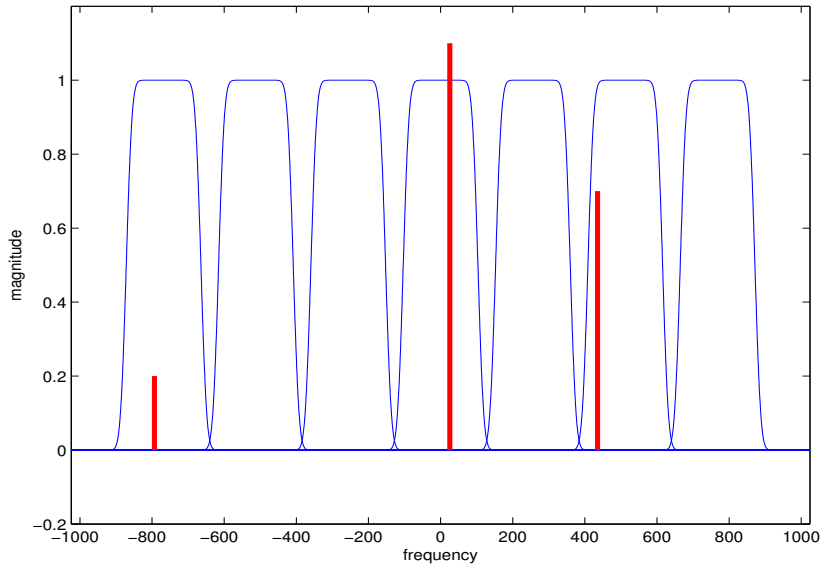
- $\hat{G}$ approximates the buckets
- G has small support

Compute $\hat{x} * \hat{G} = \widehat{(x \cdot G)}$

Reducing k-sparse recovery to 1-sparse recovery

Partition frequency space into $B = k/\alpha$ buckets for constant $\alpha \in (0,1)$





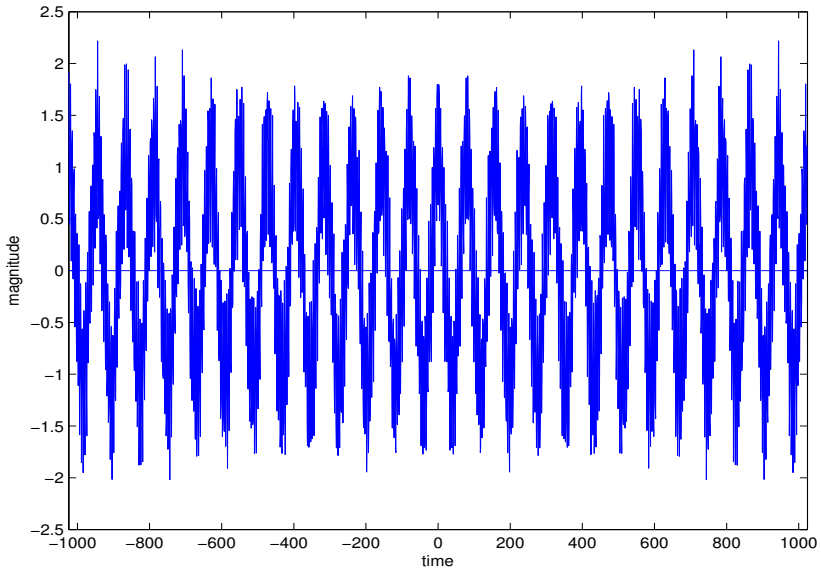
Choose a filter $G, \hat{G}$ such that

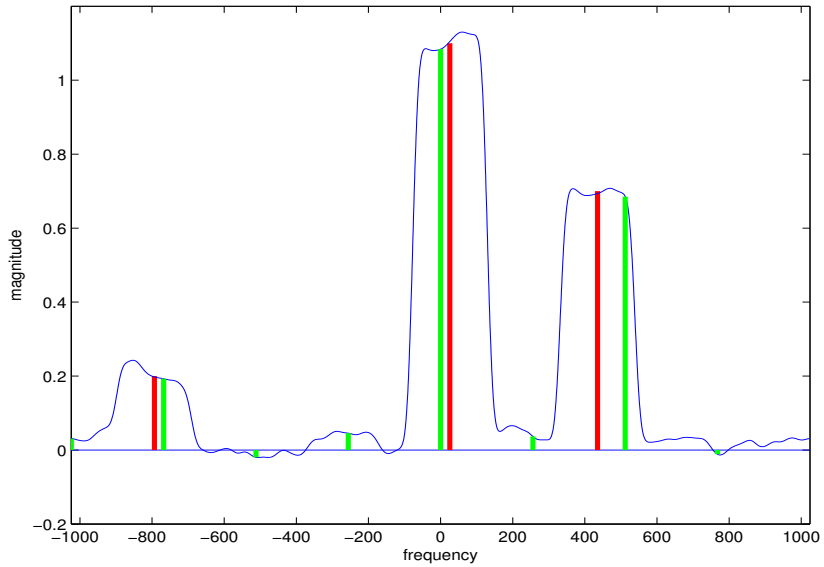
- $\hat{G}$ approximates the buckets
- G has small support

Compute $\hat{x} * \hat{G} = \widehat{(x \cdot G)}$

Reducing k-sparse recovery to 1-sparse recovery

Partition frequency space into $B = k/\alpha$ buckets for constant $\alpha \in (0,1)$





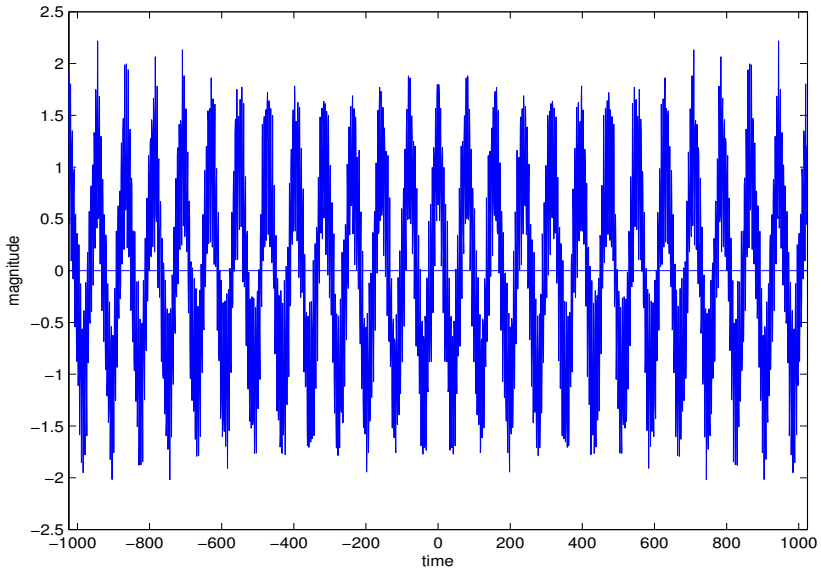
Choose a filter $G, \hat{G}$ such that

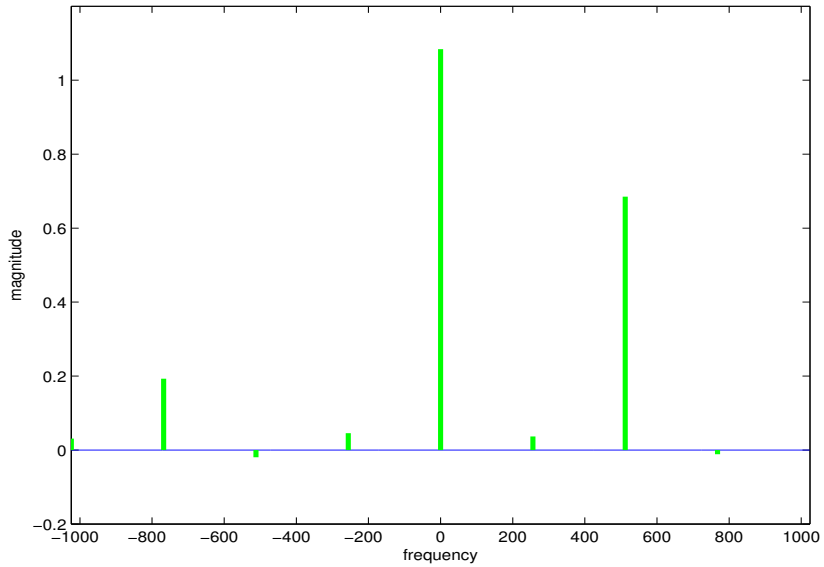
- $\hat{G}$ approximates the buckets
- G has small support

Compute $\hat{x} * \hat{G} = \widehat{(x \cdot G)}$

Reducing k-sparse recovery to 1-sparse recovery

Partition frequency space into $B = k/\alpha$ buckets for constant $\alpha \in (0,1)$





Choose a filter $G, \hat{G}$ such that

- $\hat{G}$ approximates the buckets
- G has small support

Compute $\hat{x} * \hat{G} = \widehat{(x \cdot G)}$

Sample complexity=supp G!

REFINEMENT step

REFINEMENT$(x, \widehat{y})$

- Make measurements (independent permutation+filtering)
- Locate and estimate large frequencies (1-sparse recovery)

return dominant Fourier coefficients $\widehat{z}$ of $x - y$ (approximately)

Sample complexity = support of G

REFINEMENT step

REFINEMENT$(x, \widehat{y})$

- Make measurements (independent permutation+filtering)
- Locate and estimate large frequencies (1-sparse recovery)

return dominant Fourier coefficients $\widehat{z}$ of $x - y$ (approximately)

Sample complexity = support of G

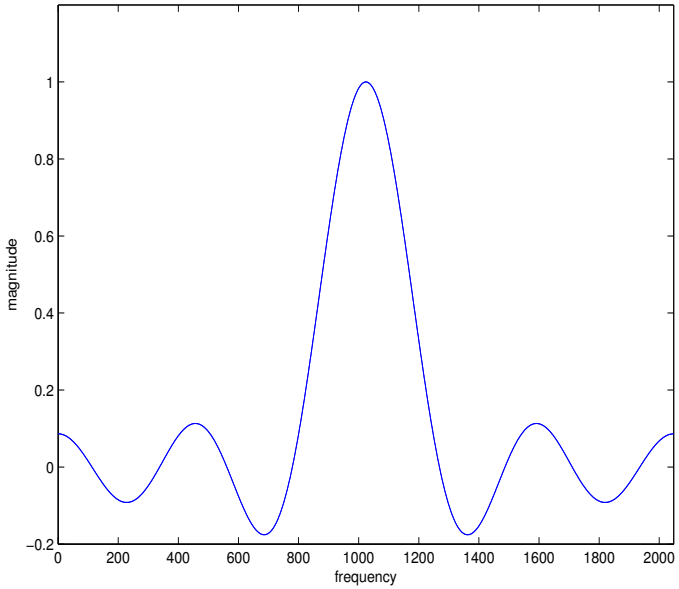
- How many measurements do we need?
- How effective is a refinement step?

Both determined by signal to noise ratio in each bucket – function of filter choice

Time domain:

support $O(k)$ [GMS'05]

Frequency domain:



SNR = $O(1)$

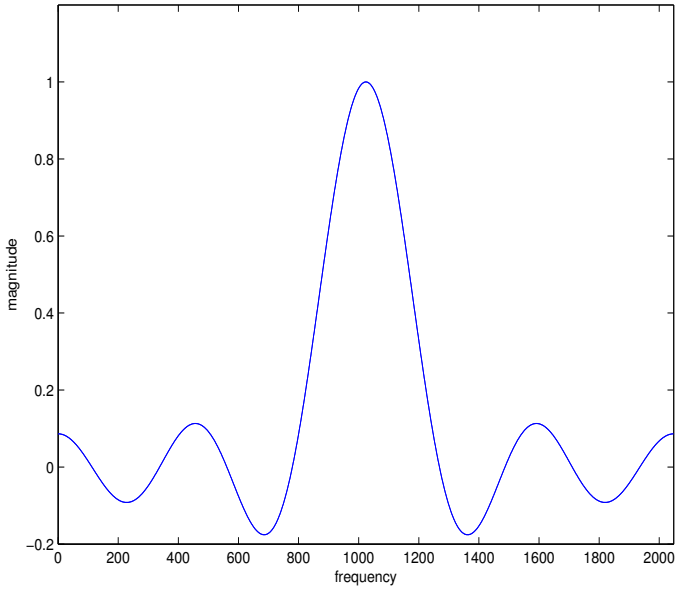
Reduce SNR by $O(1)$ factor

$\Omega(k \log^2 n)$ samples

Time domain:

support $O(k)$ [GMS'05]

Frequency domain:



SNR = $O(1)$

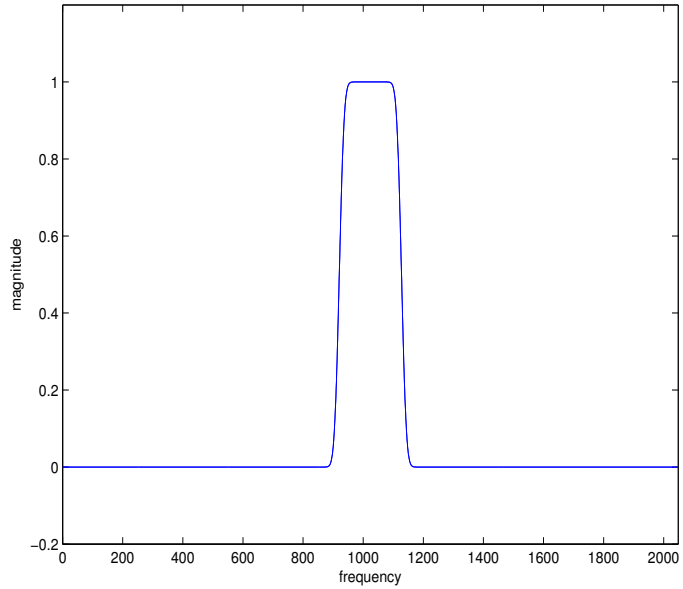
Reduce SNR by $O(1)$ factor

$\Omega(k \log^2 n)$ samples

Time domain:

support $\Theta(k \log n)$ [HIKP12]

Frequency domain:



SNR = can by poly(n)

Reduce sparsity by $O(1)$ factor

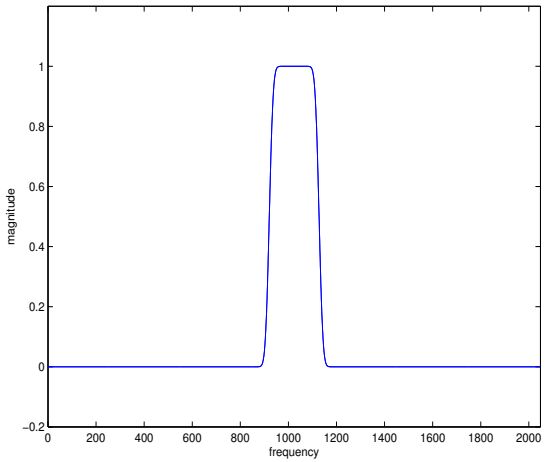
$\Omega(k \log^2 n)$ samples

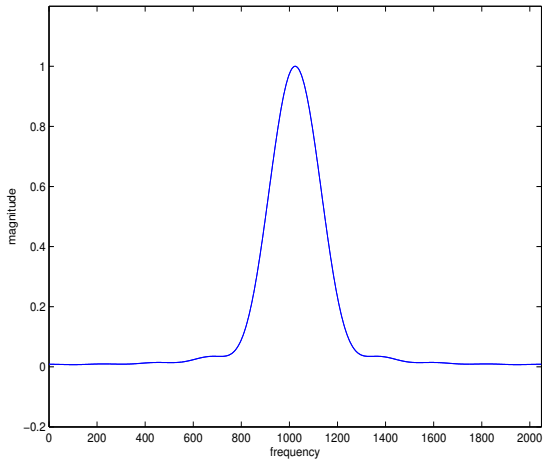
This paper: interpolate between the two extremes, get all benefits, avoid shortcomings

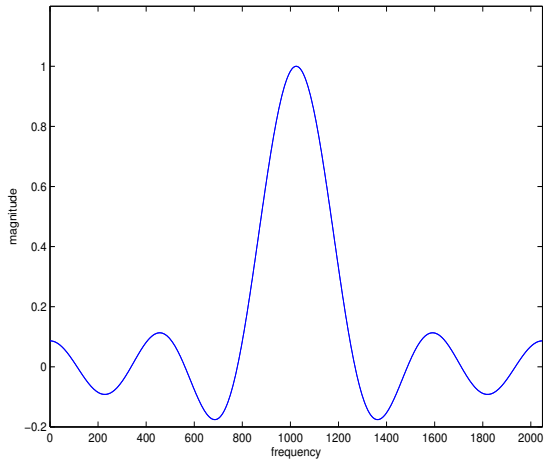
Our approach

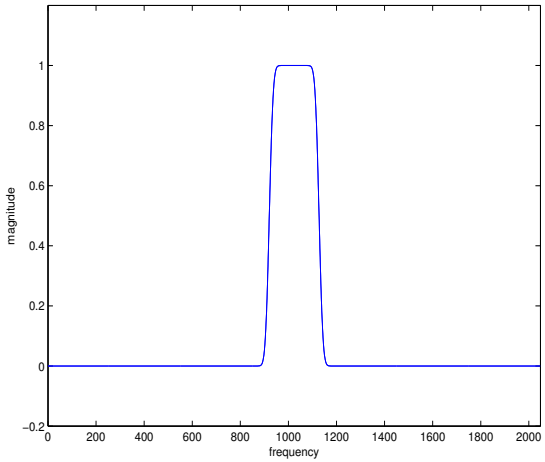
A new family of filters that adapt to current upper bound on SNR.

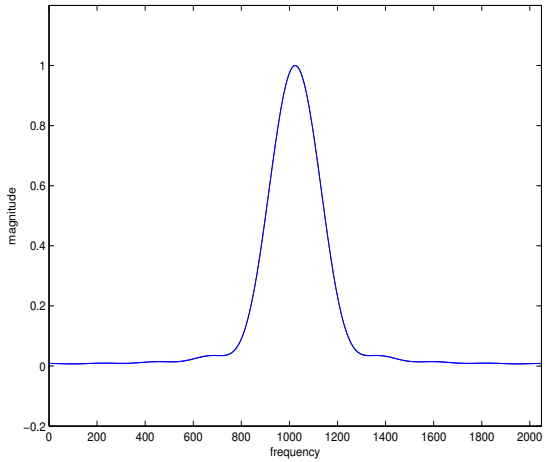
- Sharp filters initially, more blurred later

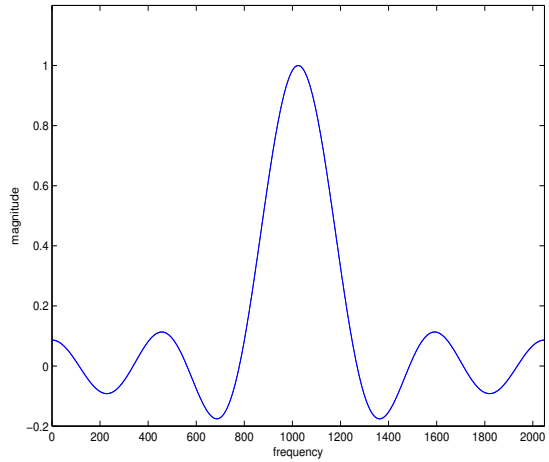






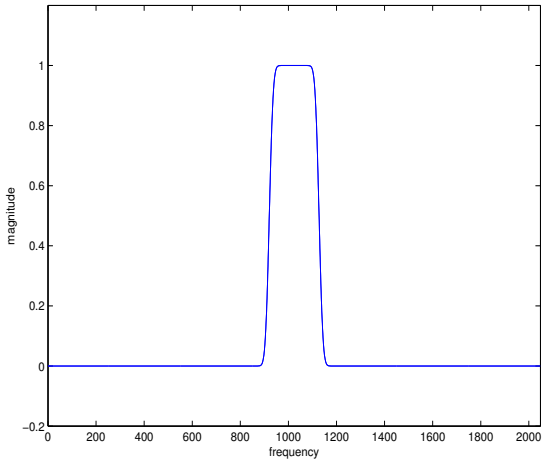


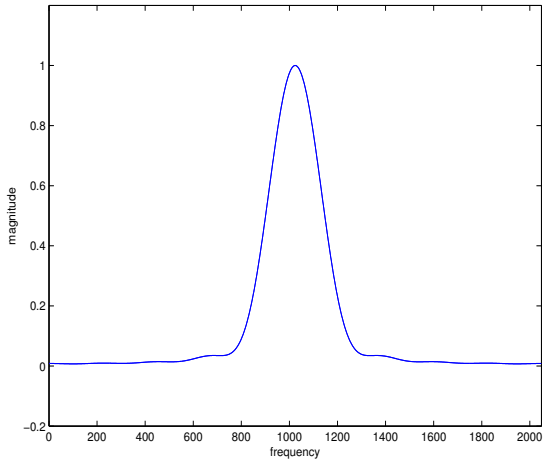


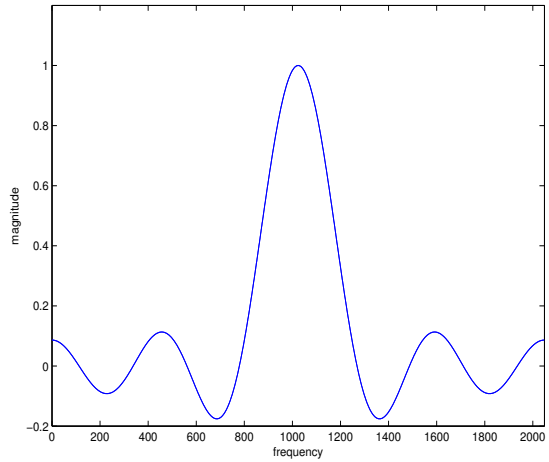


When SNR is bounded by R:

- filter support $O(k \log R)$ ($\approx$ convolve boxcar with itself $\log R$ times)







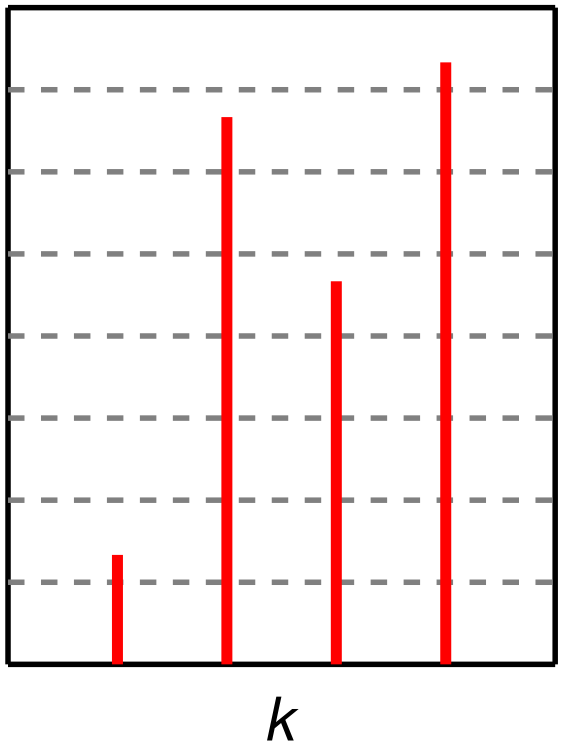
When SNR is bounded by R:

- filter support $O(k \log R)$ ($\approx$ convolve boxcar with itself $\log R$ times)
- (most) 1-sparse recovery subproblems for dominant frequencies have high SNR (about R) so $O^*(\log_R n)$ measurements!

$$O^*(k \log R \cdot \log_R n) = O^*(k \log n) \text{ samples per step!}$$

$$\underbrace{R \to R^{1/2} \to R^{1/4} \to \ldots \to C^2 \to C}_{O(\log\log n) \text{ iterations}}$$

REFINEMENT$(x, \widehat{y}, R)$



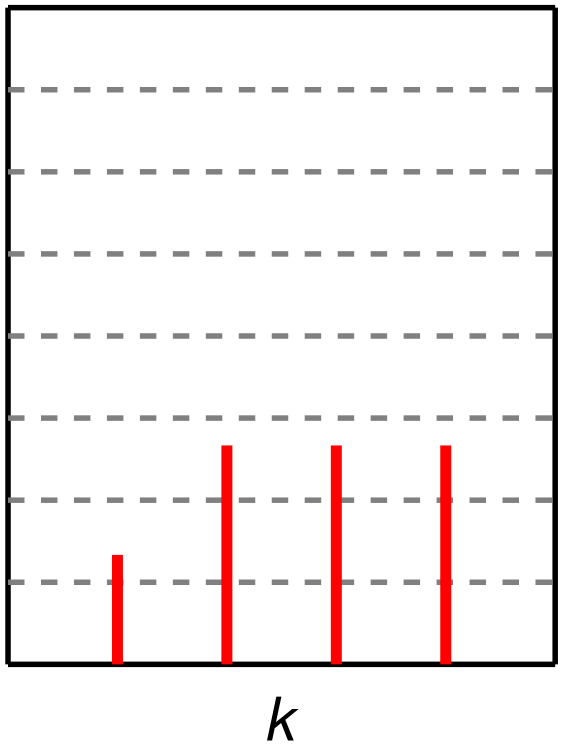
$R \cdot \mu^2$

$R^{\frac{1}{2}} \cdot \mu^2$

$\mu^2 = \text{tail noise}/B$

$$\underbrace{R \to R^{1/2} \to R^{1/4} \to \ldots \to C^2 \to C}_{O(\log\log n) \text{ iterations}}$$

REFINEMENT$(x, \widehat{y}, R)$



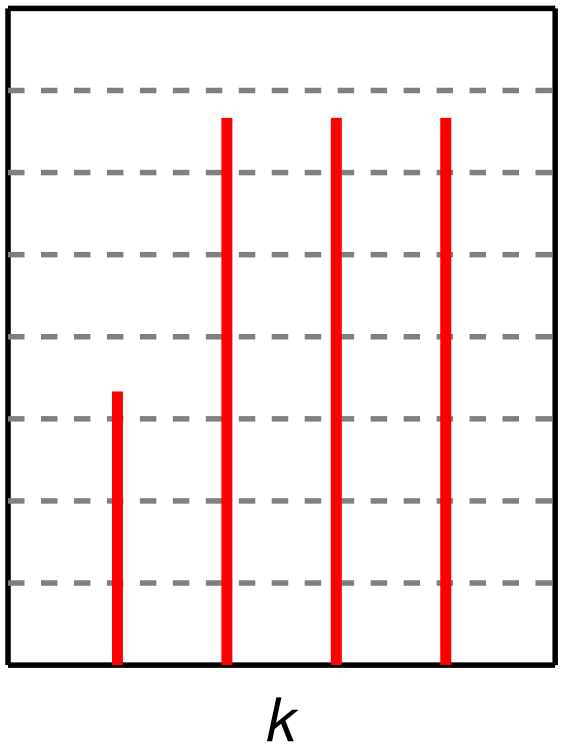
$R \cdot \mu^2$

$R^{\frac{1}{2}} \cdot \mu^2$

$\mu^2 = \text{tail noise}/B$

$$\underbrace{R \to R^{1/2} \to R^{1/4} \to \ldots \to C^2 \to C}_{O(\log\log n) \text{ iterations}}$$

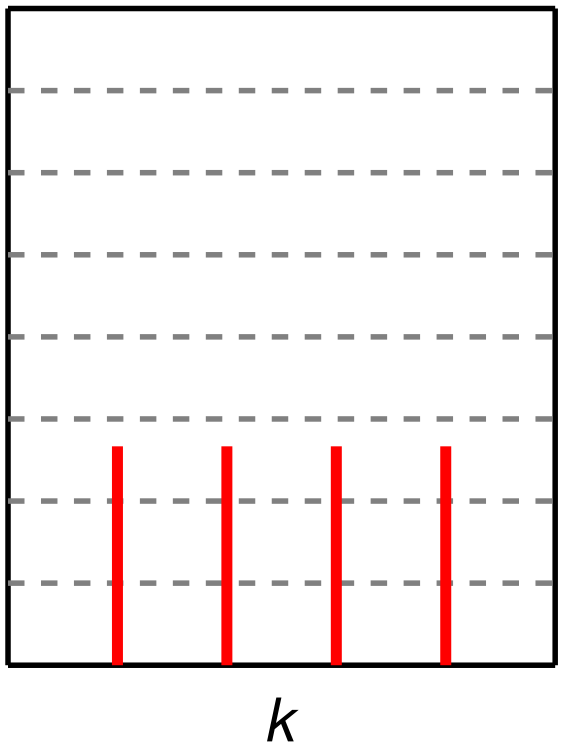
REFINEMENT$(x, \widehat{y}, R^{\frac{1}{2}})$



$\mu^2 = \text{tail noise}/B$

$$\underbrace{R \to R^{1/2} \to R^{1/4} \to \ldots \to C^2 \to C}_{O(\log\log n) \text{ iterations}}$$

REFINEMENT$(x, \hat{y}, C^2)$



$\mu^2 = \text{tail noise}/B$

Algorithm

Input: $x \in \mathbb{C}^n$
$\widehat{y}_0 \leftarrow 0$
$R_0 \leftarrow \text{poly}(n)$
For $t = 1$ to $O(\log\log n)$

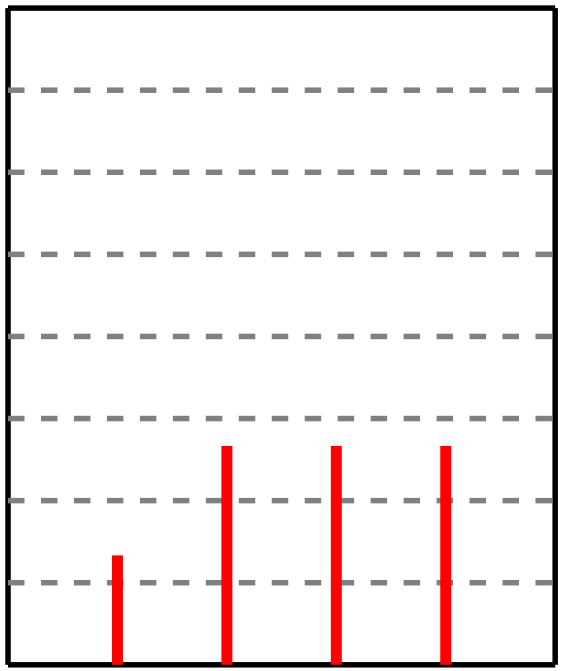
- $\widehat{z} \leftarrow \text{REFINEMENT}(x, \widehat{y}_{t-1}, R_{t-1})$ $\quad \triangleright$ Takes random samples of $x - y$
- Update $\widehat{y}_t \leftarrow \widehat{y}_{t-1} + \widehat{z}$
- $R_t \leftarrow \sqrt{R_{t-1}}$

REFINEMENT step:

- Takes $O^*(k \log n)$ samples independent of R
- Is very effective: reduces $R \rightarrow R^{\frac{1}{2}}$, so $O(\log\log n)$ iterations suffice

Refinement step analysis

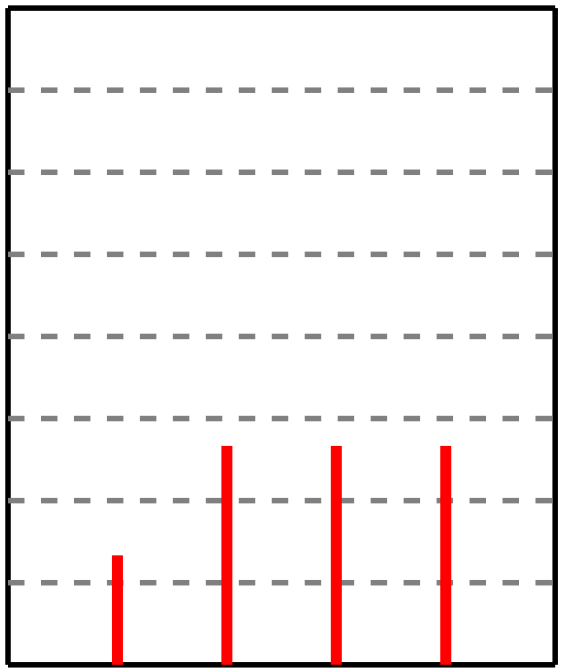
REFINEMENT$(x, \hat{y}, R)$



- Need to reduce most 'large' frequencies, i.e. $|\hat{x}_i|^2 \geq \sqrt{R}\mu^2$

Refinement step analysis

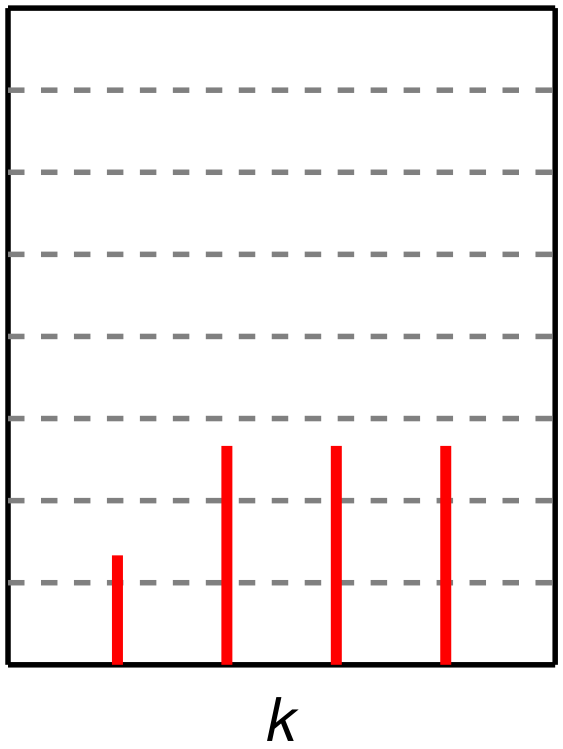
REFINEMENT$(x, \hat{y}, R)$



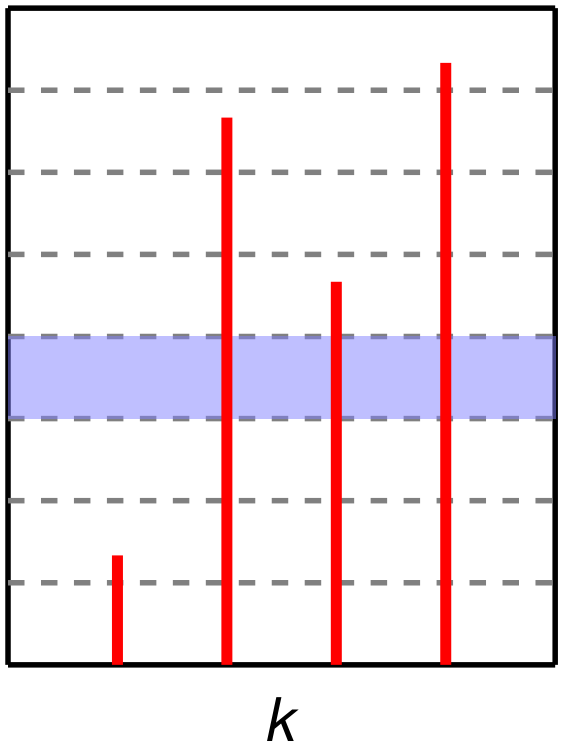
- Need to reduce most 'large' frequencies, i.e. $|\hat{x}_i|^2 \geq \sqrt{R}\mu^2$
- Most=$1 - 1/\text{poly}(R)$ fraction

Refinement step analysis

REFINEMENT$(x, \hat{y}, R)$



- Need to reduce most 'large' frequencies, i.e. $|\hat{x}_i|^2 \geq \sqrt{R}\mu^2$
- Most=$1 - 1/\text{poly}(R)$ fraction
- Iterative process, $O(\log\log n)$ steps



- partition elements into geometric weight classes
- write down recursion that governs the dynamics
- top half classes are reduced at double exponentialy rate* if we use $\Omega(\log\log R)$ levels

Conclusions

Achieved $O(k \log n (\log\log n)^C)$ samples and $O(k \log^2 n (\log\log n)^C)$, within $O((\log\log n)^C)$ of lower bound.

Recent work [IK'14?]

- optimal $O(k \log n)$ sample complexity in $\tilde{O}(n)$ time

Open questions:

- $O(k \log n)$ in $\tilde{O}(k)$ time?

Conclusions

Achieved $O(k \log n (\log\log n)^C)$ samples and $O(k \log^2 n (\log\log n)^C)$, within $O((\log\log n)^C)$ of lower bound.

Recent work [IK'14?]

- optimal $O(k \log n)$ sample complexity in $\tilde{O}(n)$ time

Open questions:

- $O(k \log n)$ in $\tilde{O}(k)$ time?

Thank you!