Lecture 2: Dealing with $k > 1$
Recap

• Have seen
  – $O(1)$-time algorithm for Fourier 1-sparse signals
  – $O(\log n \times \log \log n)$-time algorithm for Fourier 1-sparse approximation

• What about $k>1$?
What about $k>1$?

- Approach: reduce the general case to the case of $k=1$
- *Isolate* the (large) coefficients
Today: isolation via aliasing

- Needs strong assumptions to prove correctness
- But really simple and can be easily implemented
Theorem: Assume $L$ divides $n$. Let $a'=(a_0, a_L, a_{2L}, \ldots)$ be a vector with $n/L$ entries. Then

$$\hat{a}'_u = \sum_{l=0}^{L-1} \hat{a}_{u+n/L \cdot l}$$
Proof

• See e.g.,:
  http://www.dsprelated.com/dspbooks/mdft/Downsampling_Theorem_Aliasing_Theorem.html
  (somewhat different notation and scaling)
Theorem: Suppose that $\hat{a}$ is generated by summing up $k$ coefficients in positions selected independently and uniformly at random from $0 \ldots n-1$. I.e.,

$$\hat{a} = \sum_{t=1}^{k} b_t \delta_{j(t)}$$

where $b_t$ fixed and $j(1) \ldots j(k)$ i.i.d. random variables chosen uniformly from $0 \ldots n-1$.

Then, given $a$, we can recover $\hat{a}$ in time $O(k^2 \log k)$ with probability $>1/2$.

**Average case result**
Algorithm+Analysis

• Algorithm:
  – Can assume $k^2 << n$, otherwise FFT
  – Set $L = n/k^2$
  – Define:
    • $a' = (a_0, a_L, a_{2L}, \ldots)$ of length $n/L = k^2$
    • $a'' = (a_1, a_{L+1}, a_{2L+1}, \ldots)$ of length $n/L = k^2$
  – Compute $\hat{a}'$ and $\hat{a}''$, where
    $$\hat{a}'_u = \sum_{l=0}^{L-1} \hat{a}_{u+n/L} l$$
    $$\hat{a}''_u = \sum_{l=0}^{L-1} \hat{a}_{u+n/L} l \omega^{u+n/L}$$
  – For each non-zero $\hat{a}'_u$ apply the two-point sampling approach to $\hat{a}'_u$ and $\hat{a}''_u$
    • From $\hat{a}''_u/\hat{a}'_u$ can recover $u+n/L$, the index of the non-zero coordinate

• Analysis:
  – The probability that there exists $u$ s.t. the sum defining $\hat{a}'_u$ contains $>1$ non-zero terms is at most
    $$k(k-1)/2 * 1/L < 1/2$$
Reducing $k^2$ to $k$

• Bin using **two co-prime** aliasing filters
  – Same frequencies don’t collide in two filters
  – Need to assume that $n=p\times q$, $k<<\ min(p,q)$
• Identify isolated freq. in one filter and subtract them from the other; and iterate
• More in the next lecture