Nearly Linear-Time Algorithms for Structured Sparsity

Piotr Indyk

MIT CSAIL

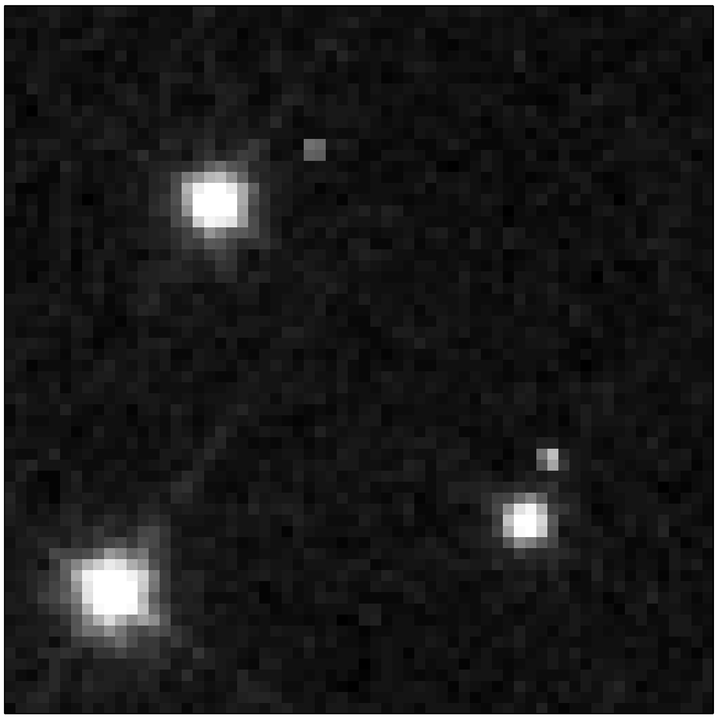
Joint work with C. Hegde and L. Schmidt (MIT),
J. Kane and L. Lu and X. Chi and D. Hohl (Shell)

But first….

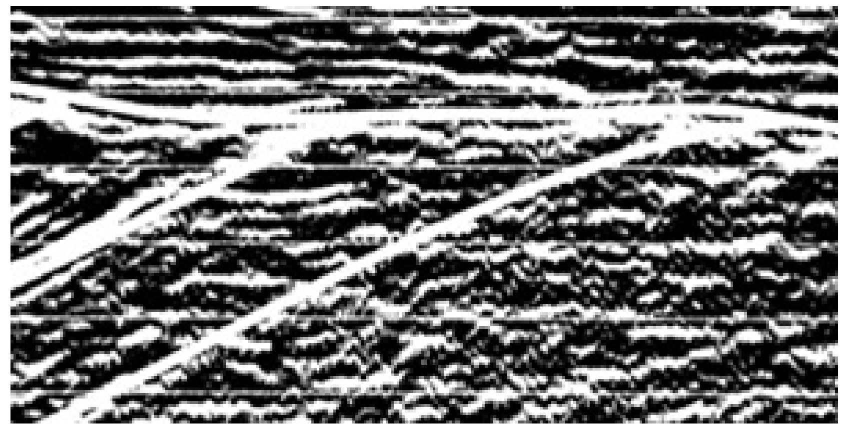

Congratulations Boston!
Snowiest Season On Record
108.6 INCHES!
Previous Record 107.6” (1995-1996)
Website: www.weather.gov/boston
Twitter: @NWSBoston
Facebook: http://www.facebook.com/NWSBoston

Sparsity in data

- Data is often **sparse**



Hubble image
(cropped)



seismic image

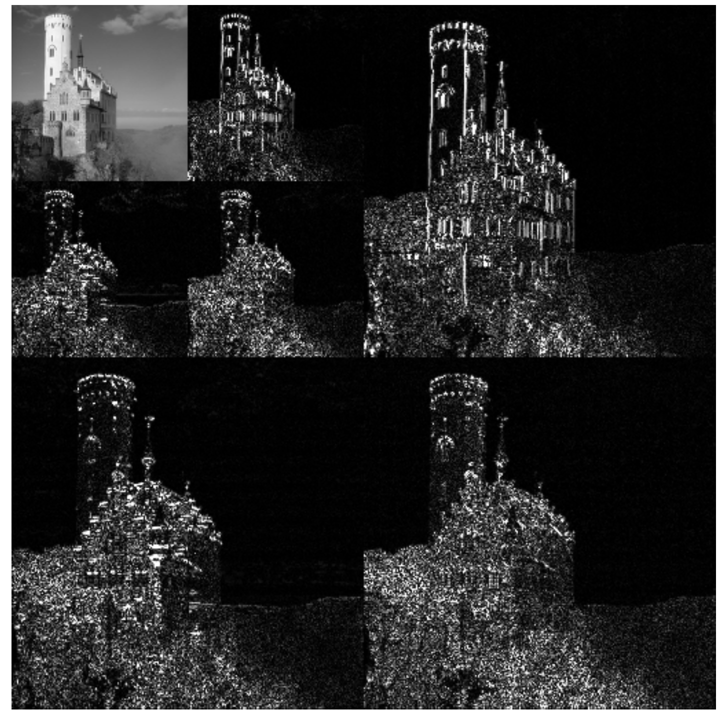
Data can be specified by values and locations of their *k* large coefficients (*2k* numbers)

Sparsity in data

- Data is often **sparsely expressed** using a suitable linear transformation

n pixels

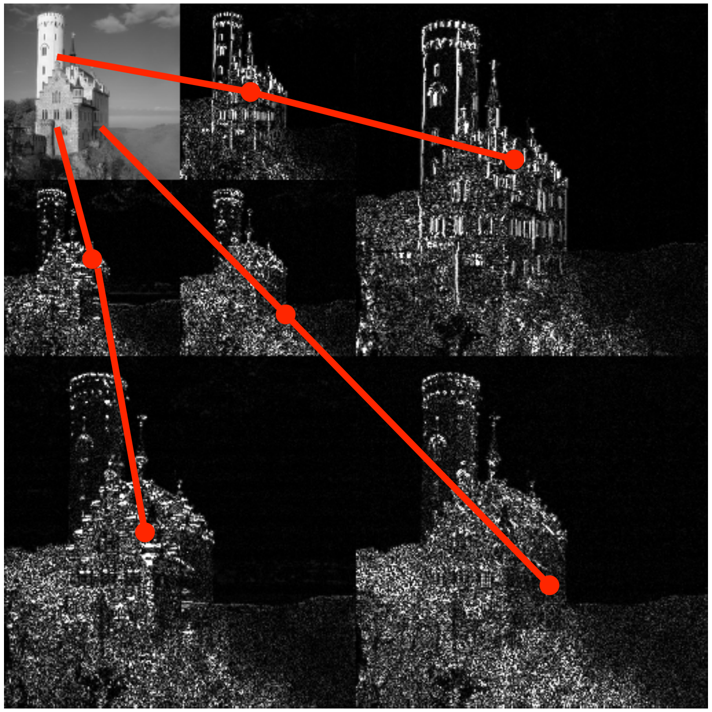
Wavelet transform



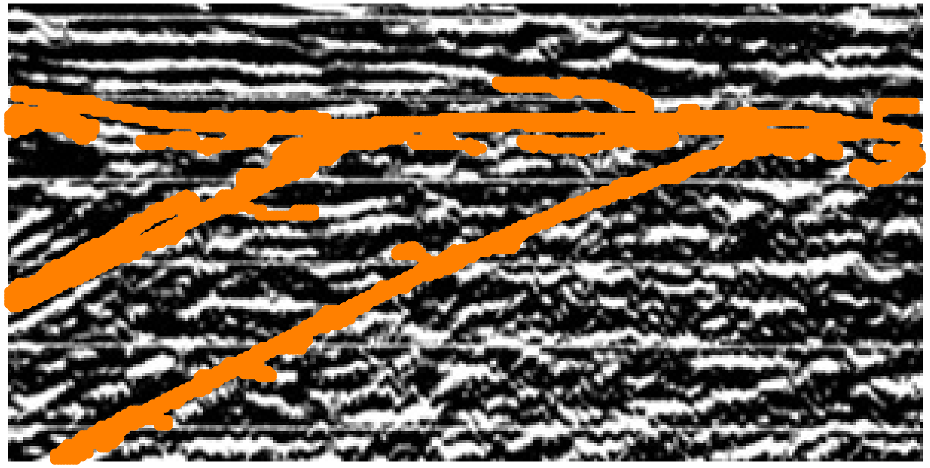
$k \ll n$ large wavelet coefficients

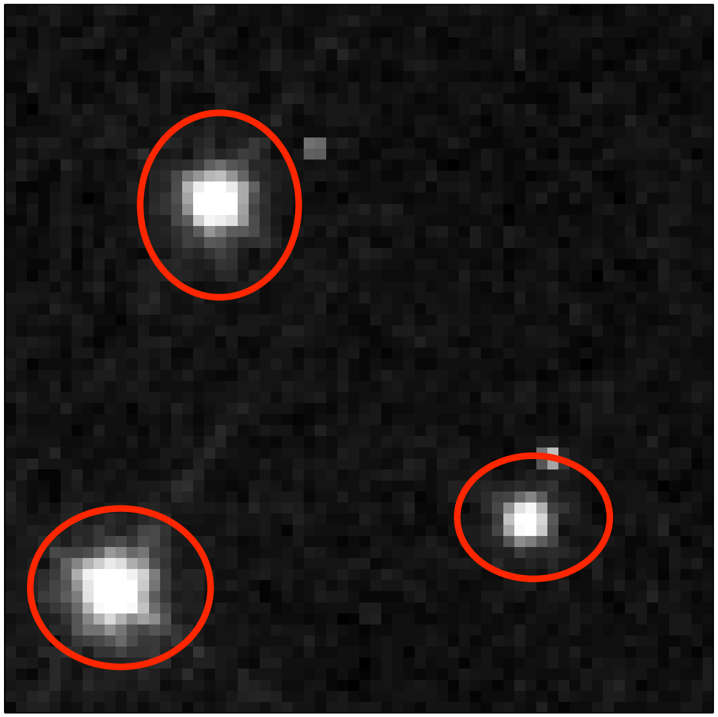
Data can be specified by values and locations of their *k* large **wavelet** coefficients (*2k* numbers)

Beyond sparsity



- Notion of sparsity captures **simple primary structure**
- But locations of large coefficients often exhibit **rich secondary structure**





This talk

- Structured sparsity:
  - Models
  - Examples: Block sparsity,Tree sparsity, Constrained EMD, Clustered Sparsity
- Efficient algorithms: how to extract structured sparse representations quickly
- Applications:
  - (Approximation-tolerant) model-based compressive sensing
  - Fault detection in seismic images

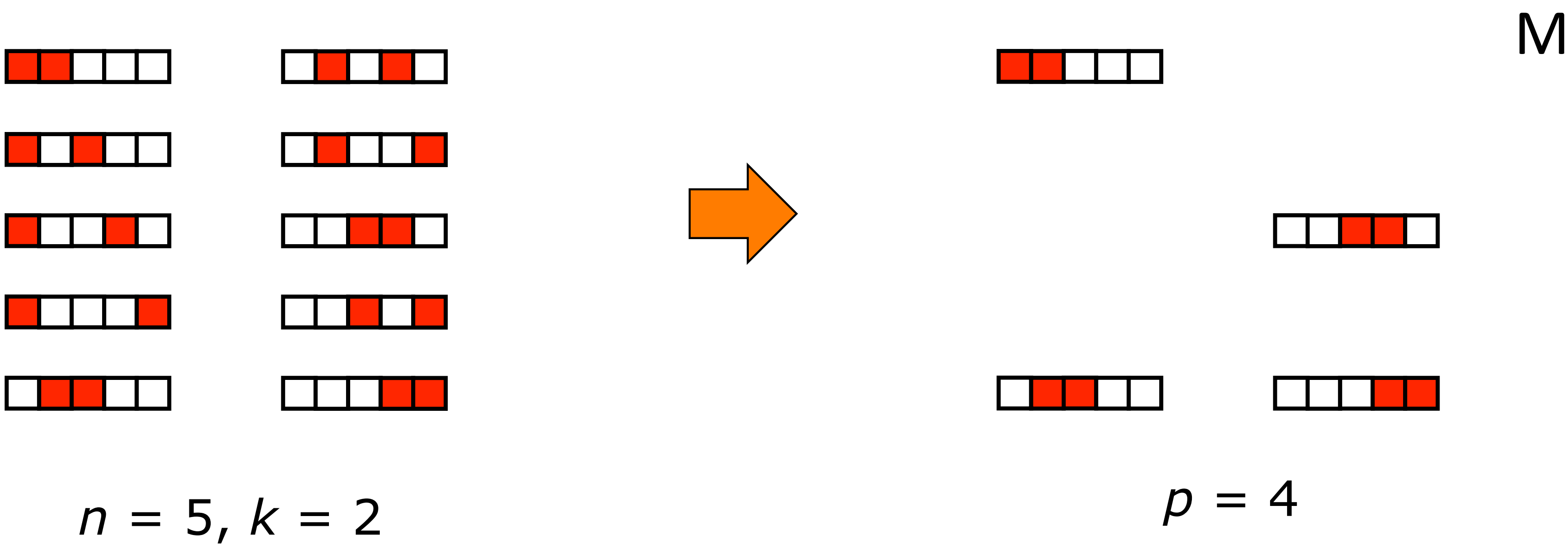
Modeling approach

Def: Specify a list of p allowable sparsity patterns

$M = \{\Omega_1, \dots, \Omega_p\}$ where $\Omega_i \subseteq [n]$, $|\Omega_i| \leq k$

Then, a **structured sparsity model** is the space of signals supported on one of the patterns in M

$$\mathcal{M} = \{x \in \mathbb{R}^n \mid \exists\, \Omega_i \in \Omega : \operatorname{supp}(x) \subseteq \Omega_i\}$$

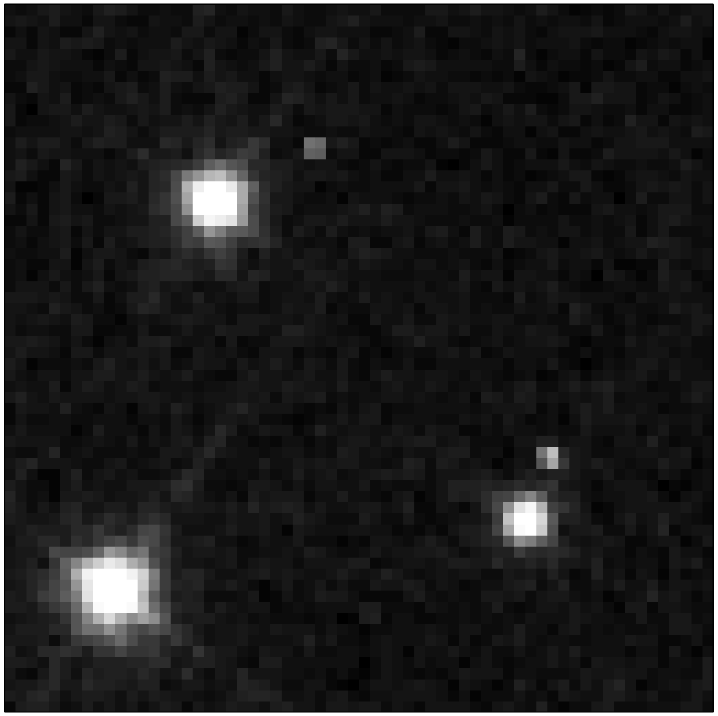


Model I: Block sparsity

- "Large coefficients hang out in groups"
- Parameters: k, b (block length) and ℓ (number of blocks)
- The range {1…n} is partitioned into b-length blocks $B_1 \ldots B_{n/b}$
- M contains all combinations of ℓ blocks, i.e.,

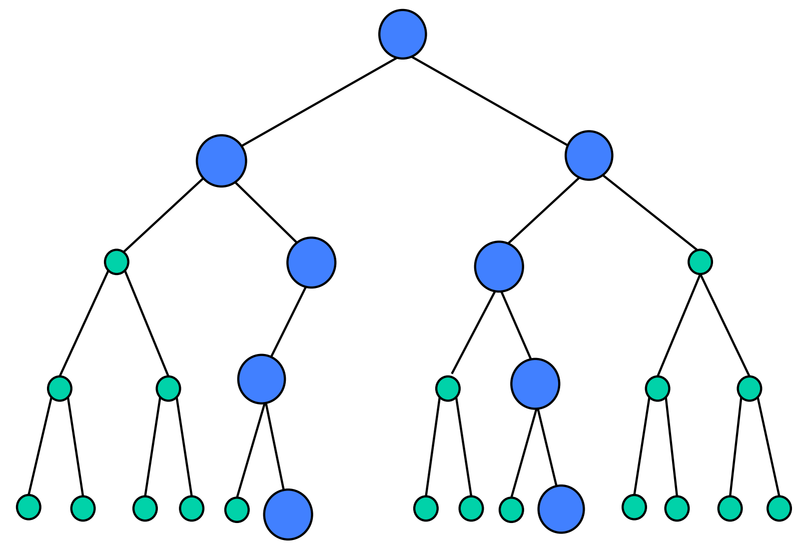
$$M=\{ B_{i1} \cup \ldots \cup B_{i\ell} : i_1, .., i_\ell \in \{1..n/b\} \}$$

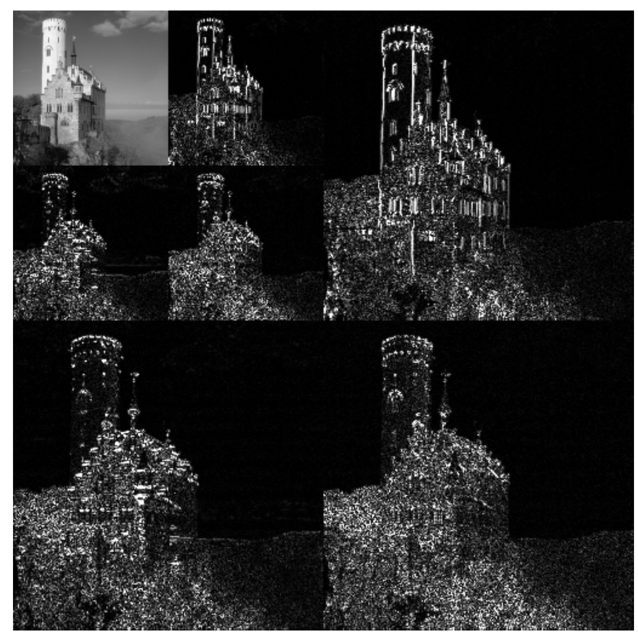
- Sparsity $k=b\ell$



Model II: Tree-sparsity

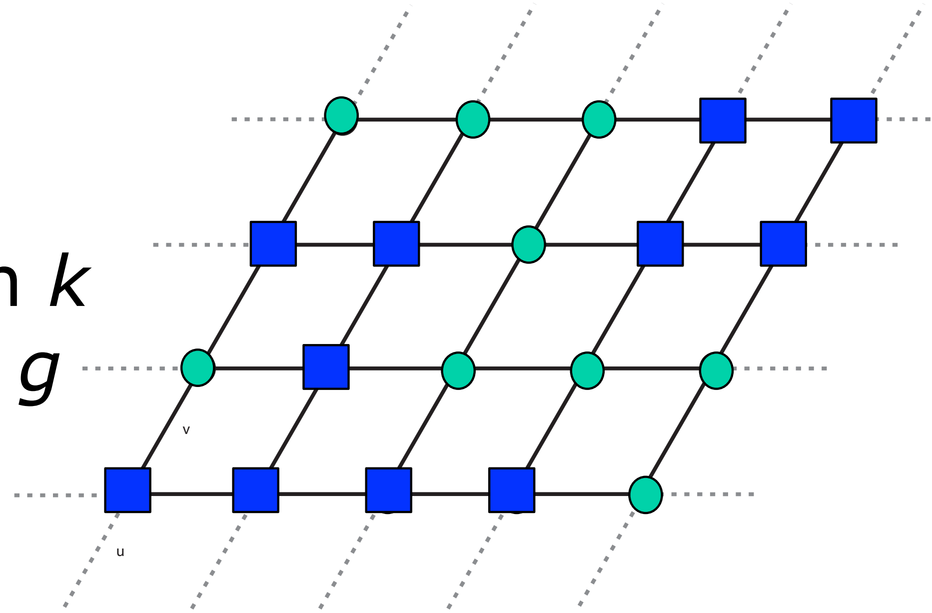
- "Large coefficients hang out on a tree"
- Parameters: k,t
- Coefficients are nodes in a full t-ary tree
- M is the set of all rooted connected subtrees of size k

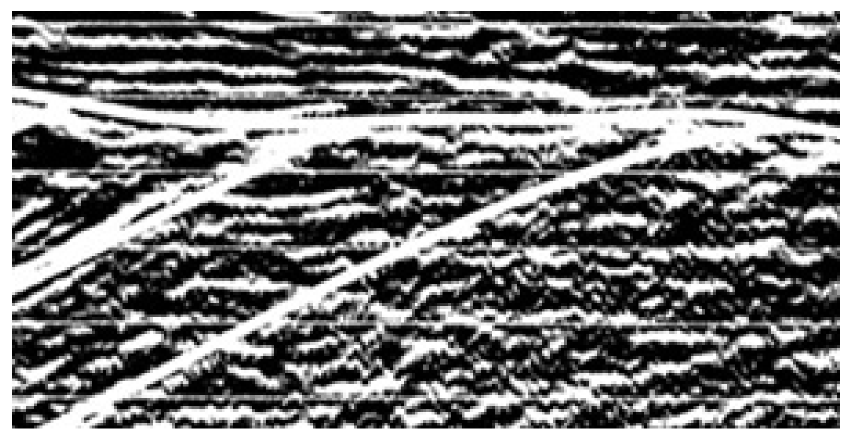




Model III: Graph sparsity

- Parameters: k, g, graph G
- Coefficients are nodes in G
- M contains all subgraphs with *k* nodes that are clustered into *g* *connected components*

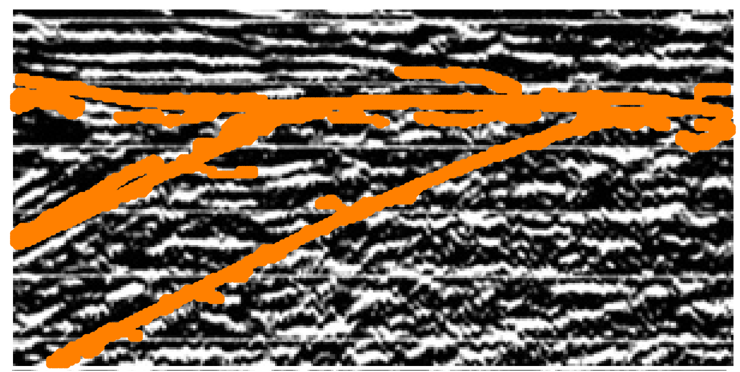




What can we do with those models ?

- Structured sparsity model specifies a *hypothesis class* for signals of interest
- For an arbitrary input signal x, a model projection oracle **extracts structure** by returning the "closest" signal in model

$$M(x) = \text{argmin}_{\Omega \in M} ||x - x_\Omega||_2$$

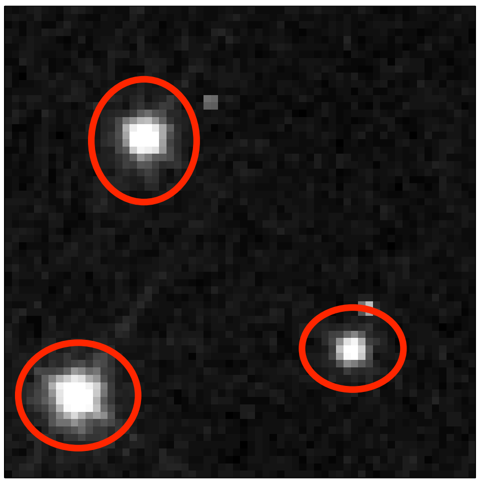


- Applications:
  - Compression
  - Denoising
  - Machine learning
  - Model-based compressive sensing
  - …

Algorithms for model projection

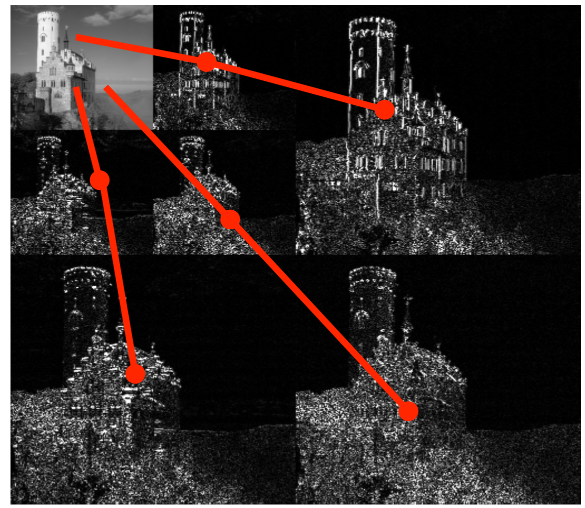
- Good news: several important models admit projection oracles with **polynomial time complexity**

Blocks



Block thresholding
(**linear** time: O(n))

Trees



Dynamic programming
(**rectangular** time: O(nk))

- Bad news:
  - Polynomial time is not enough. E.g., consider a 'moderate' problem: n = 10 million, k = 5% of n. Then, $nk >$ **5×10^{12}**
  - For some models (e.g., graph sparsity), model projection is NP-hard

Approximation to the rescue

- Instead of finding an exact solution to the projection

  $$M(x) = \text{argmin}_{\Omega \in M} ||x-x_\Omega||_2$$

  we solve it approximately (and much faster)
- What does "approximately" mean ?
  - (Tail) $||x-T(x)|| \leq C_T \text{ argmin}_{\Omega \in M} ||x-x_\Omega||_2$
  - (Head) $||H(x)|| \geq C_H \text{ argmax}_{\Omega \in M} ||x_\Omega||_2$
- Choice depends on applications
  - Tail: works great if approximation is good
  - Head: meaningful output even if approximation is not good
- For compressive sensing application we need both !

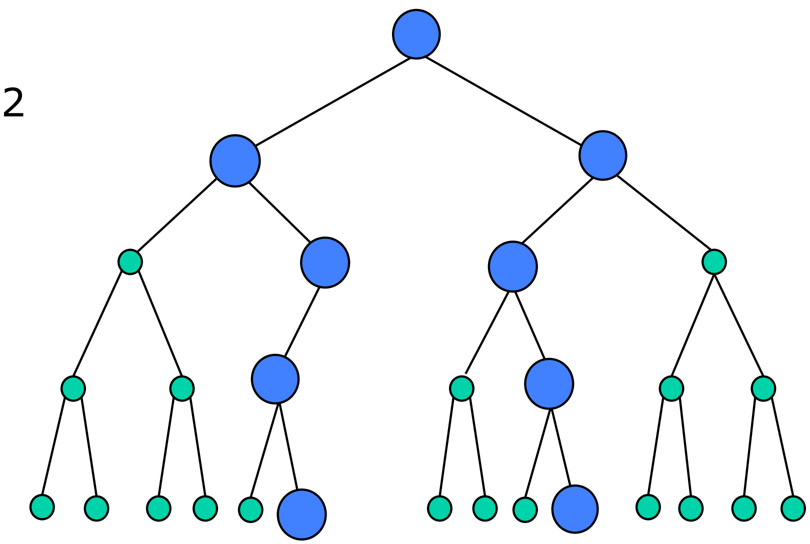
Our results

Model	Previous time	Our time
Tree sparsity	O(nk) [exact]	O(n $\log^2 n$) [H/T]
Graph sparsity	O(n^τ) [approximate]	O(n $\log^4 n$) [H/T]
Constrained EMD		

Tree sparsity

(Tail) $||x-T(x)|| \leq C_T \text{ argmin}_{\Omega \in \text{Tree}} ||x-x_\Omega||_2$

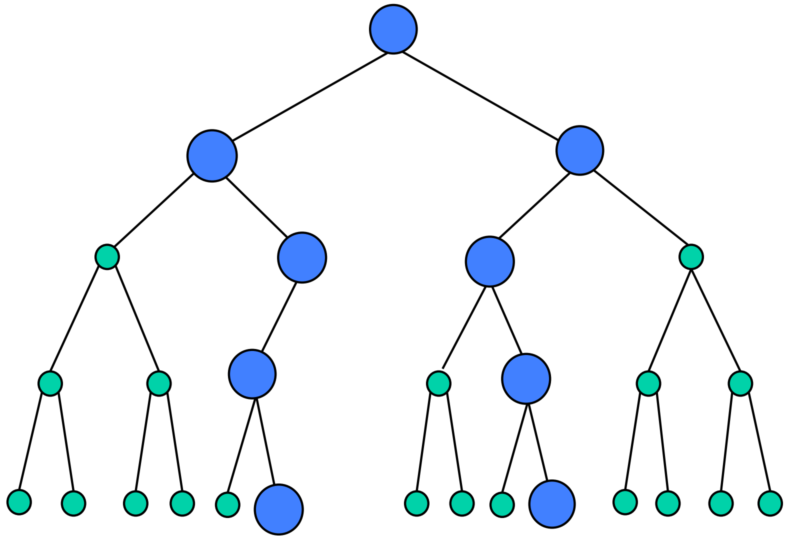
(Head) $||H(x)|| \geq C_H \text{ argmax}_{\Omega \in \text{Tree}} ||x_\Omega||_2$



	Runtime	Guarantee
Baraniuk-Jones ‘94	O(n log n)	?
Donoho ‘97	O(n)	?
Bohanec-Bratko ‘94	$O(n^2)$	Exact
Cartis-Thompson ‘13	O(nk)	Exact
This work	O(n log n)	**Approx. Head**
This work	$O(n \log n + k \log^2 n)$	**Approx. Tail**

Proof (techniques)

- Approximate "tail" oracle:
  - Idea: **Lagrangian relaxation** + Pareto curve analysis
- Approximate "head" oracle:
  - Idea: **Submodular maximization**



Implication for compressive sensing

Let x be a k-sparse vector in R^n that belongs to one of the aforementioned models*. There is a matrix A with O(k) rows s.t. given Ax+e, we can recover x* such that

$$||x-x^*||_2 \leq ||e||_2$$

in time roughly

$$\log n * (n \log^{O(1)} n + \text{matrix-vector-mult-time})$$

* Assuming constant degree, number of components <k/log n

Experiments: 2D images

Original image

Least-squares

Sparsity

$n = 512 \times 512$
$k \sim 10{,}000$
$m \sim 35{,}000$

$m/n = 12\%$

Tree structure (exact)

Tree structure (approx)

Experiments: Speed

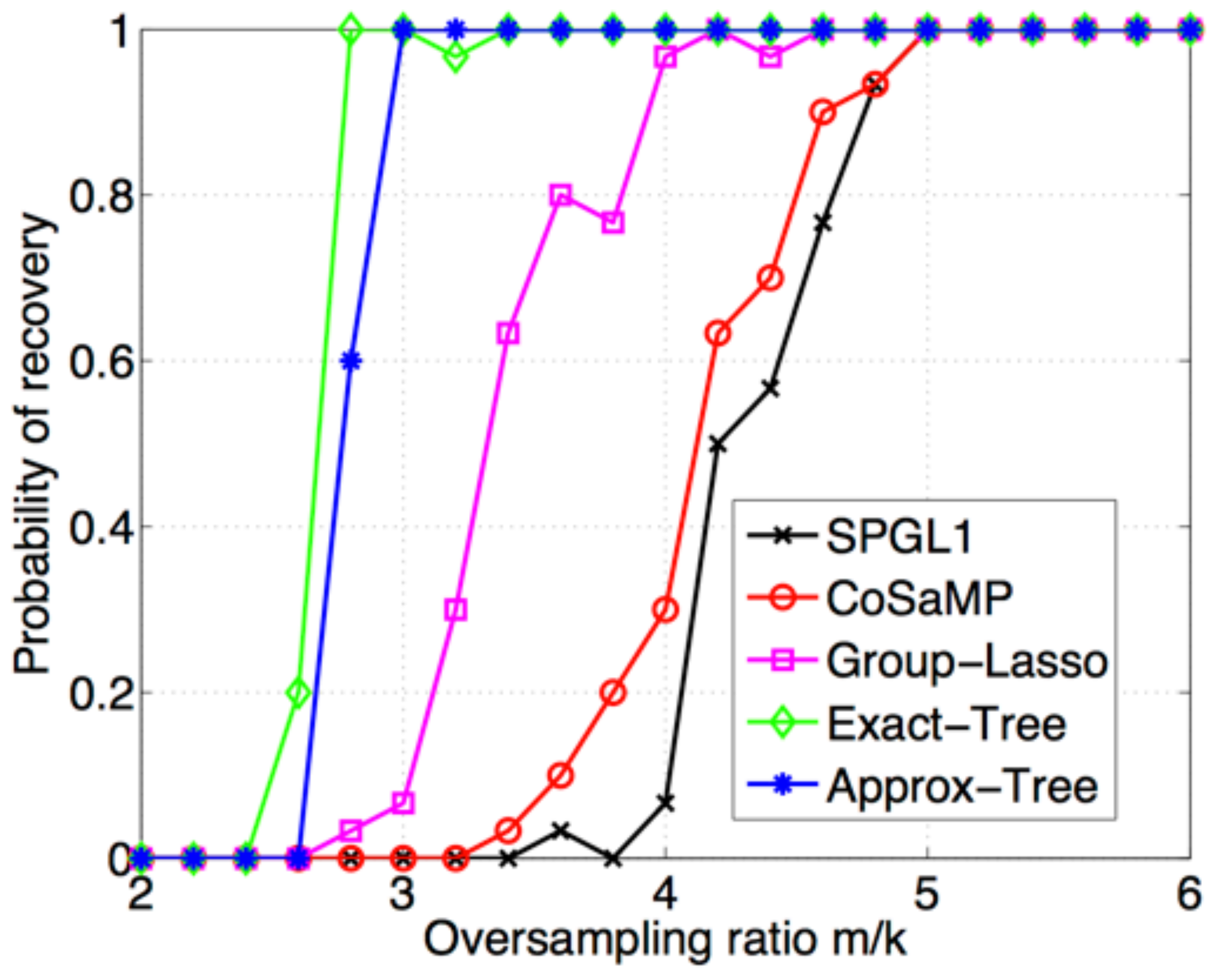
Test instance: 512 x 512 image.

Algorithm	**Exact**	**Approximate**	**2 Matlab FFTs**
Runtime	4.4175 sec	0.0109 sec	0.0150 sec

* **~400x speedup** over exact (dynamic programming based) model-projection for trees

* **Efficient algorithms** for tree-structured data modeling

Phase Transition



- Test signals of length n=1024 that is k=41 sparse in the wavelet domain
- Random Gaussian measurements (noiseless)
- Success is defined as recovering the signal within relative Euclidean norm error of 5%

Conclusions/Open Problems

- Approximation algorithms for structured sparsity
  - Rich collection of interesting algorithmic questions
  - Applications (compressive sensing, applications, etc)
- Open questions:
  - Fast and provable matrices A
    - Recall: time log n*(n$\log^{O(1)}$ n + matrix-vector-mult-time))
    - In theory we are using Gaussian matrices, which are provable but slow
    - In practice we are using Fourier matrices, which are fast but heuristic

Acknowledgments and references

- Images:
  - Boston Snowman- National Weather Service Boston.
  - Hubble telescope image
  http://heritage.stsci.edu/gallery/bwgallery/bw0405/index.shtml
  - Seismic image: "Structural framework of Southeastern Malay Basin", Ngah, 2000.
  - Chicago skyline
  http://news.uic.edu/files/2014/11/DG11_09_07_082_sm.jpg
- References:

  - Chinmay Hegde, Piotr Indyk, Ludwig Schmidt: Nearly Linear-Time Model-Based Compressive Sensing. ICALP 2014.
  - Chinmay Hegde, Piotr Indyk, Ludwig Schmidt: A fast approximation algorithm for tree-sparse recovery. ISIT 2014.
  - Chinmay Hegde, Piotr Indyk, Ludwig Schmidt: Approximation-Tolerant Model-Based Compressive Sensing. SODA 2014.
  - Ludwig Schmidt, Chinmay Hegde, Piotr Indyk, Jonathan Kane, Ligang Lu, Detlef Hohl: Automatic fault localization using the generalized Earth Mover's distance. ICASSP 2014.