Compressive Sensing of Streams of Pulses

Chinmay Hegde and *Richard Baraniuk*

Rice University

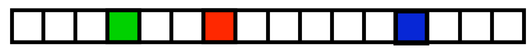
Concise Signal Model: Sparsity

- **Sparse** signal:

  - only K out of N coordinates nonzero

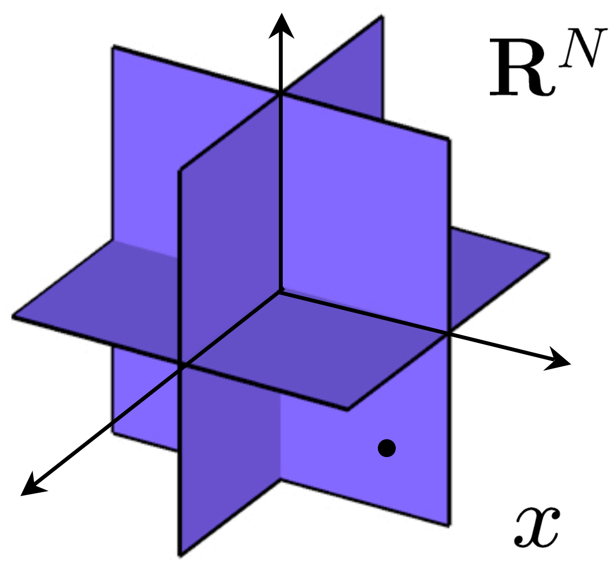
Concise Signal Model: Sparsity

- **Sparse** signal:



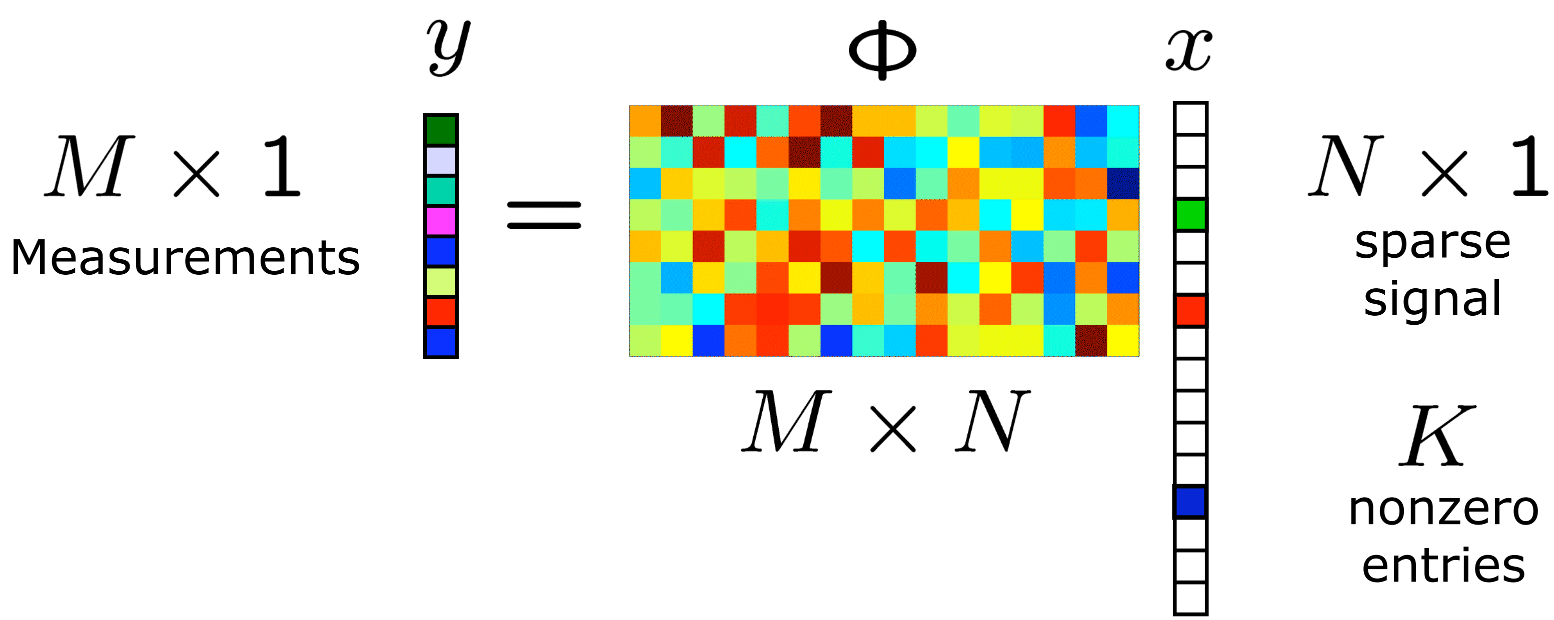
  – only *K* out of *N* coordinates nonzero

- Geometry: *union* of *K*-dimensional subspaces aligned w/ coordinate axes



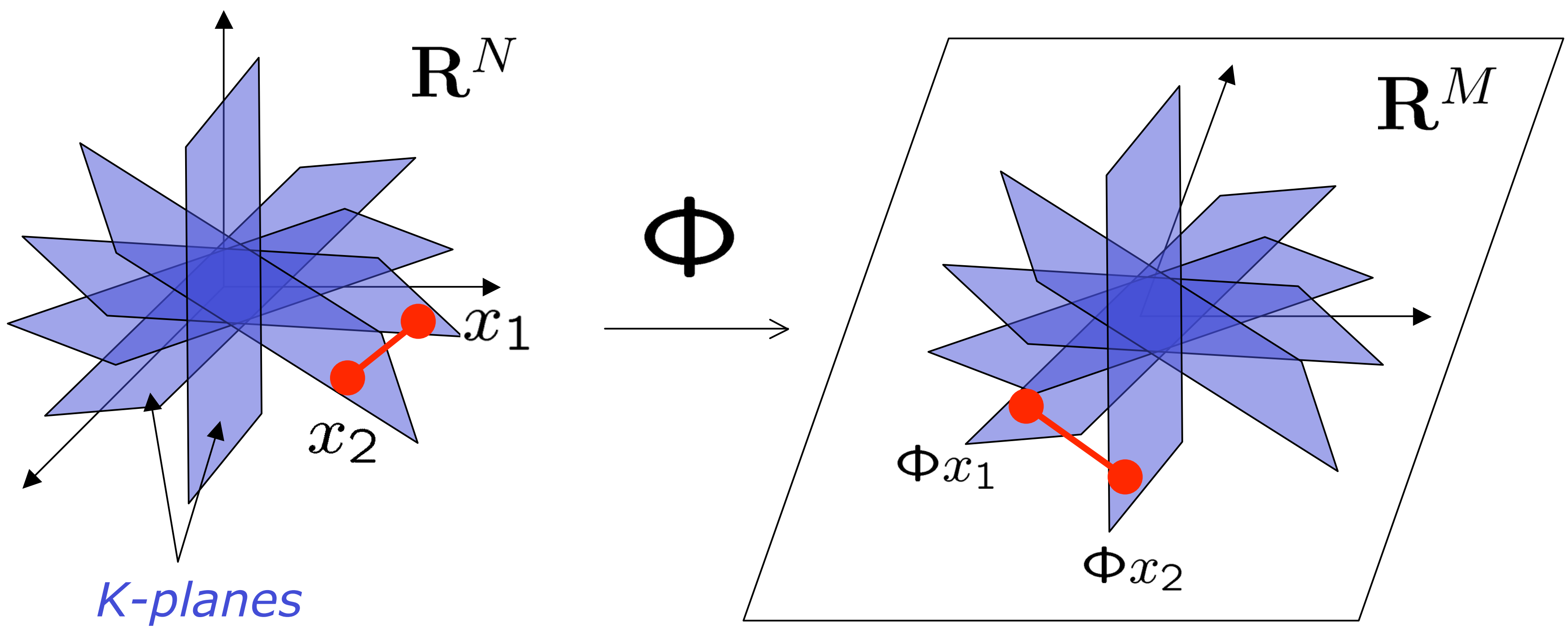
Compressive Sensing

- **Sampling** via dimensionality reduction

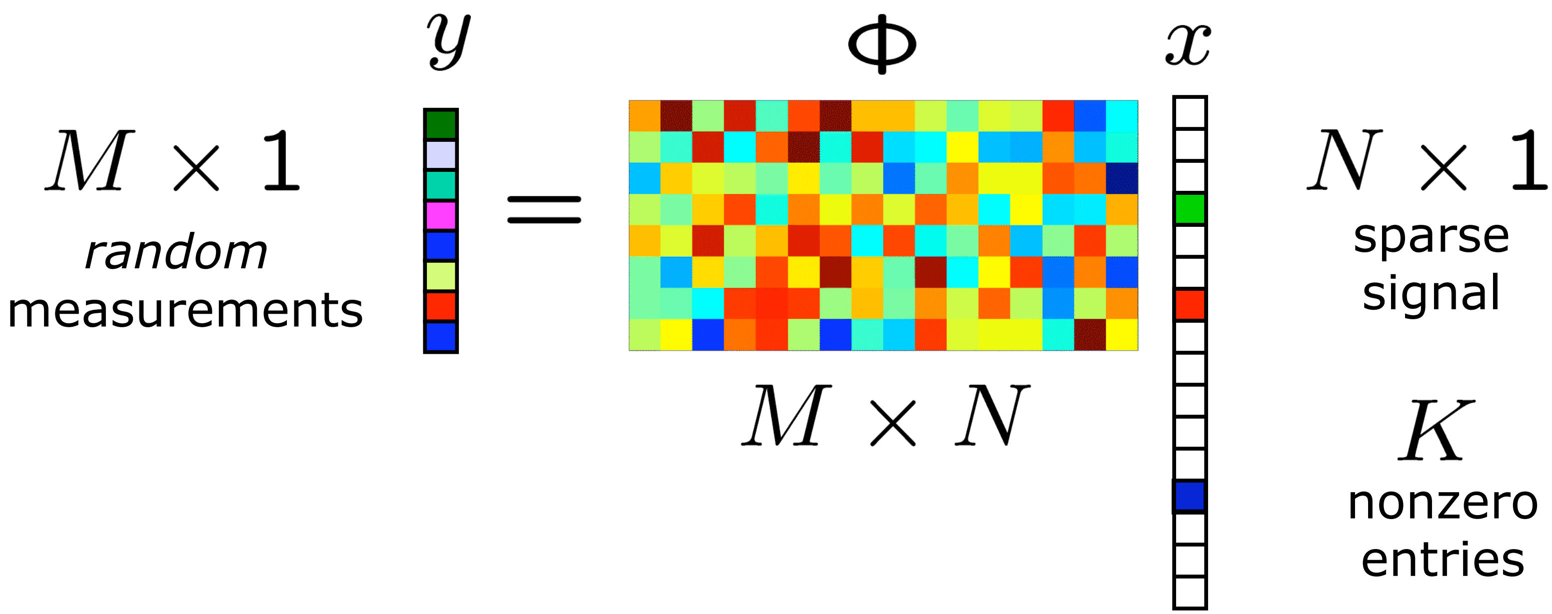


Restricted Isometry Property (RIP)

- Preserve the structure of sparse signals



Compressive Sensing

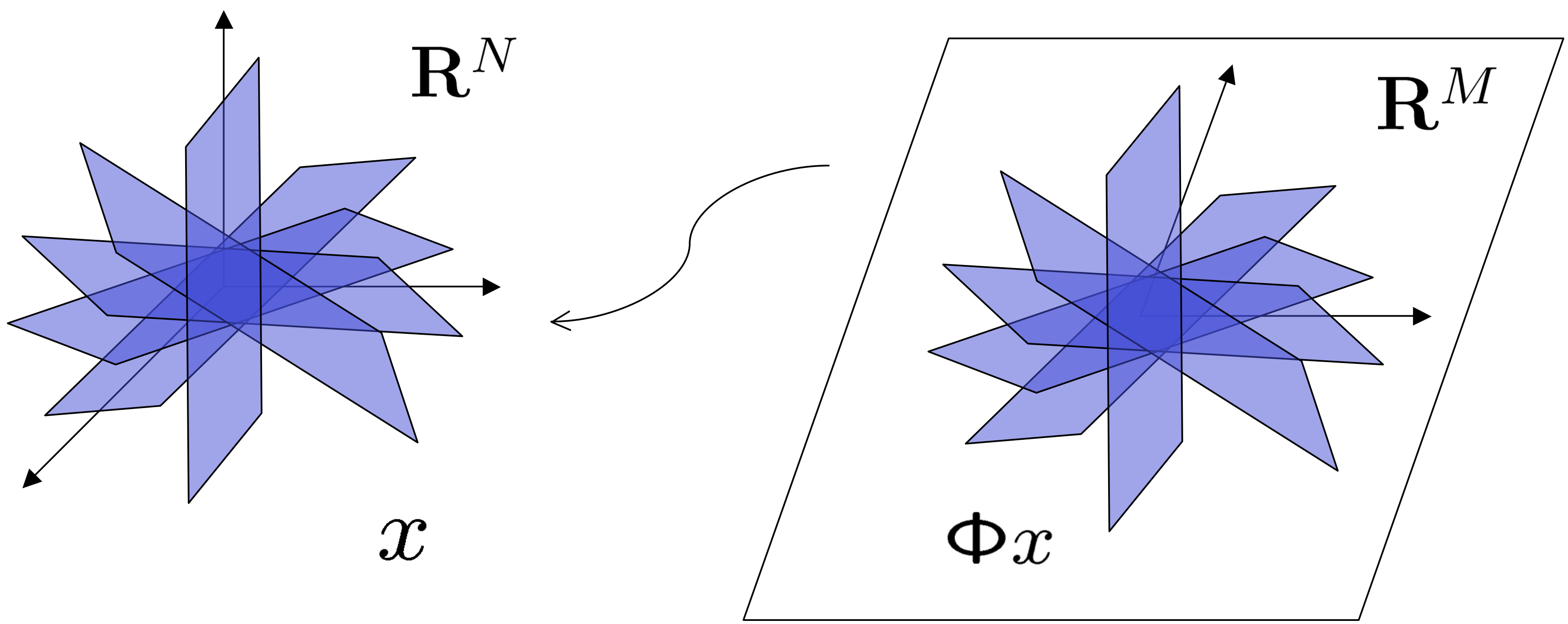


- *Random* subgaussian matrix Φ has the **RIP** whp if

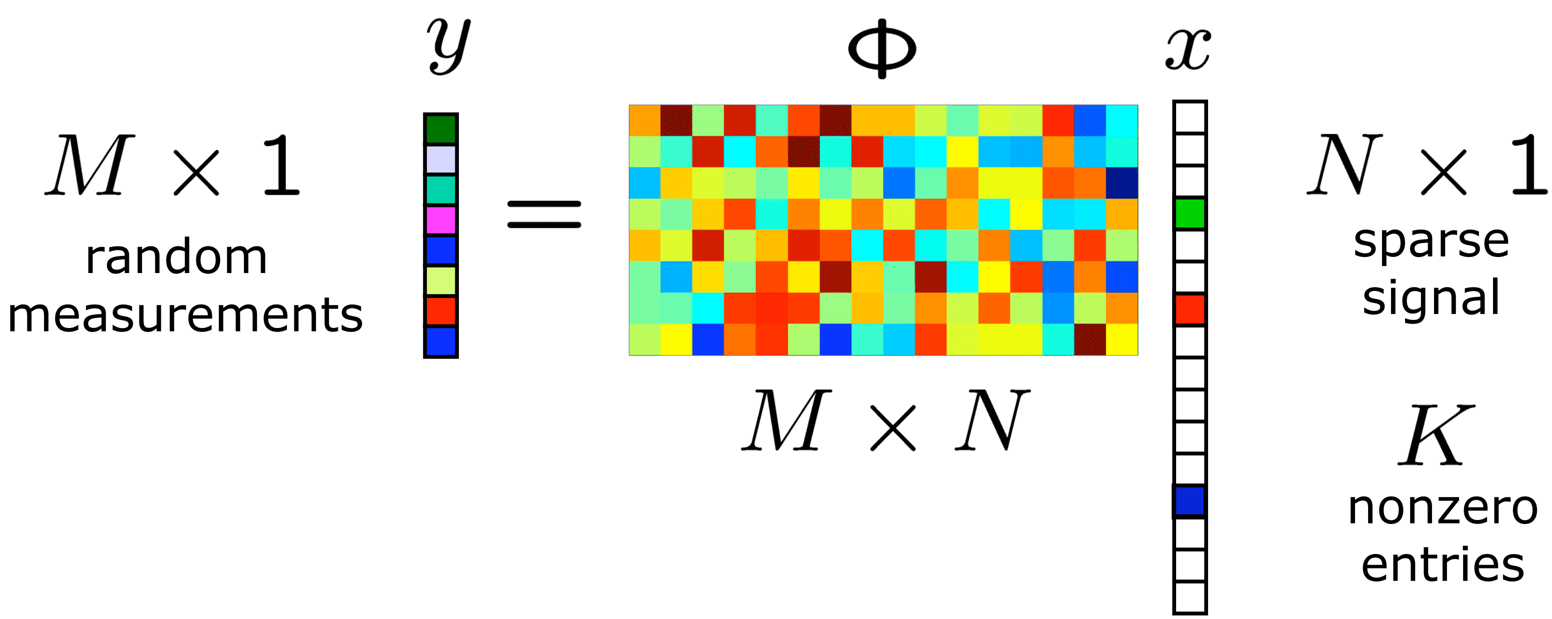
$$M = O(K + \log \binom{N}{K})$$

Stable Recovery

- Efficient, stable algorithms that give back signal



Compressive Sensing



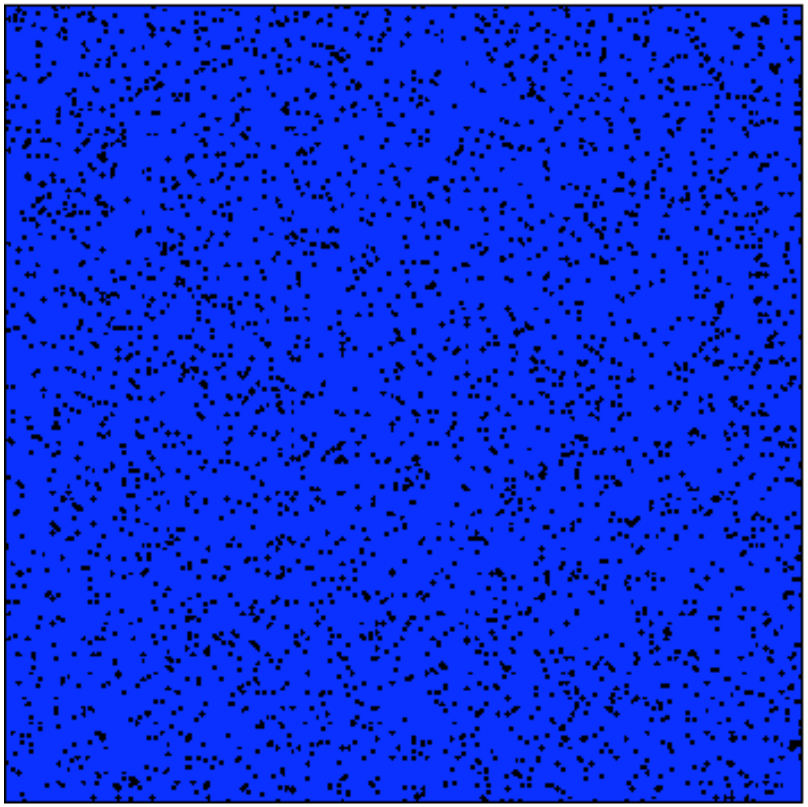
- ℓ_1-optimization
  [C, R, T]; [D]; [F,W,N]; [H,Y,Z]

- Greedy algorithms
  - OMP [G, T]
  - iterated thresholding [N, F]; [D, D, DeM]; [B, D]
  - CoSaMP [N,T]; Subspace Pursuit [D,M]

Beyond Sparse Models

Beyond Sparse Models

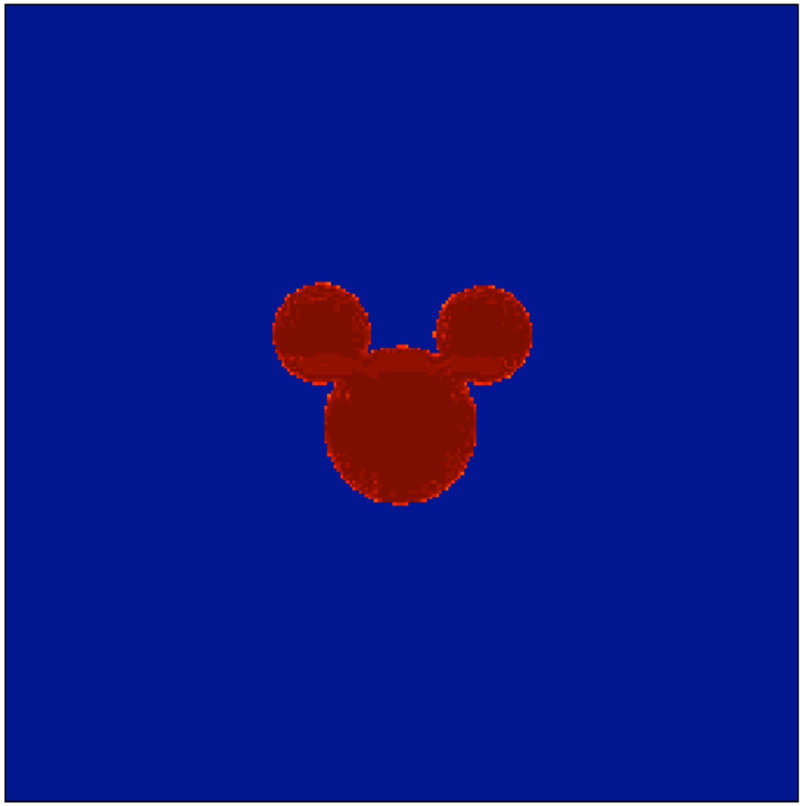
- Sparsity captures **simplistic primary structure**



5% sparse image

Beyond Sparse Models

- Most real-world apps exhibit **additional structure**



5% sparse image

Model-based CS

- ***K*-sparse structured sparsity model** comprises a *reduced* set of *K*-dim canonical subspaces

- **Model-RIP:** stable embedding
  [B, D]; [B,D,DeV,W]

$$M = O(K + \log(m_K))$$

- **Recovery:** simple modification of iterative support selection algorithms

Model-based CS

- ***K*-sparse *structured sparsity model*** comprises a *reduced* set of *K*-dim canonical subspaces

- **Model-RIP:** stable embedding
  [B, D]; [B,D,DeV,W]

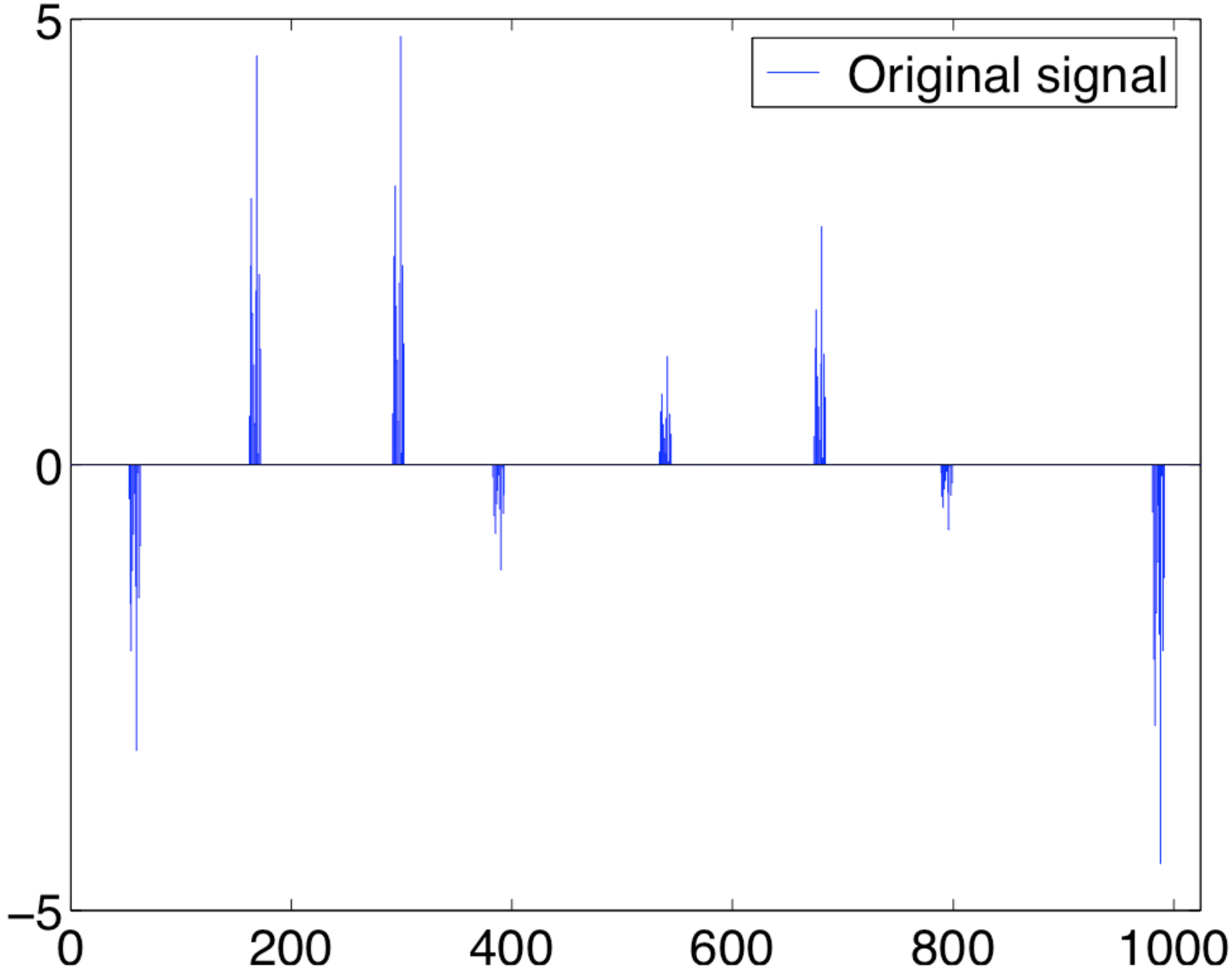
$$M = O(K + \log(m_K))$$

- **Recovery:** simple modification of iterative support selection algorithms

- **Models are good!!**

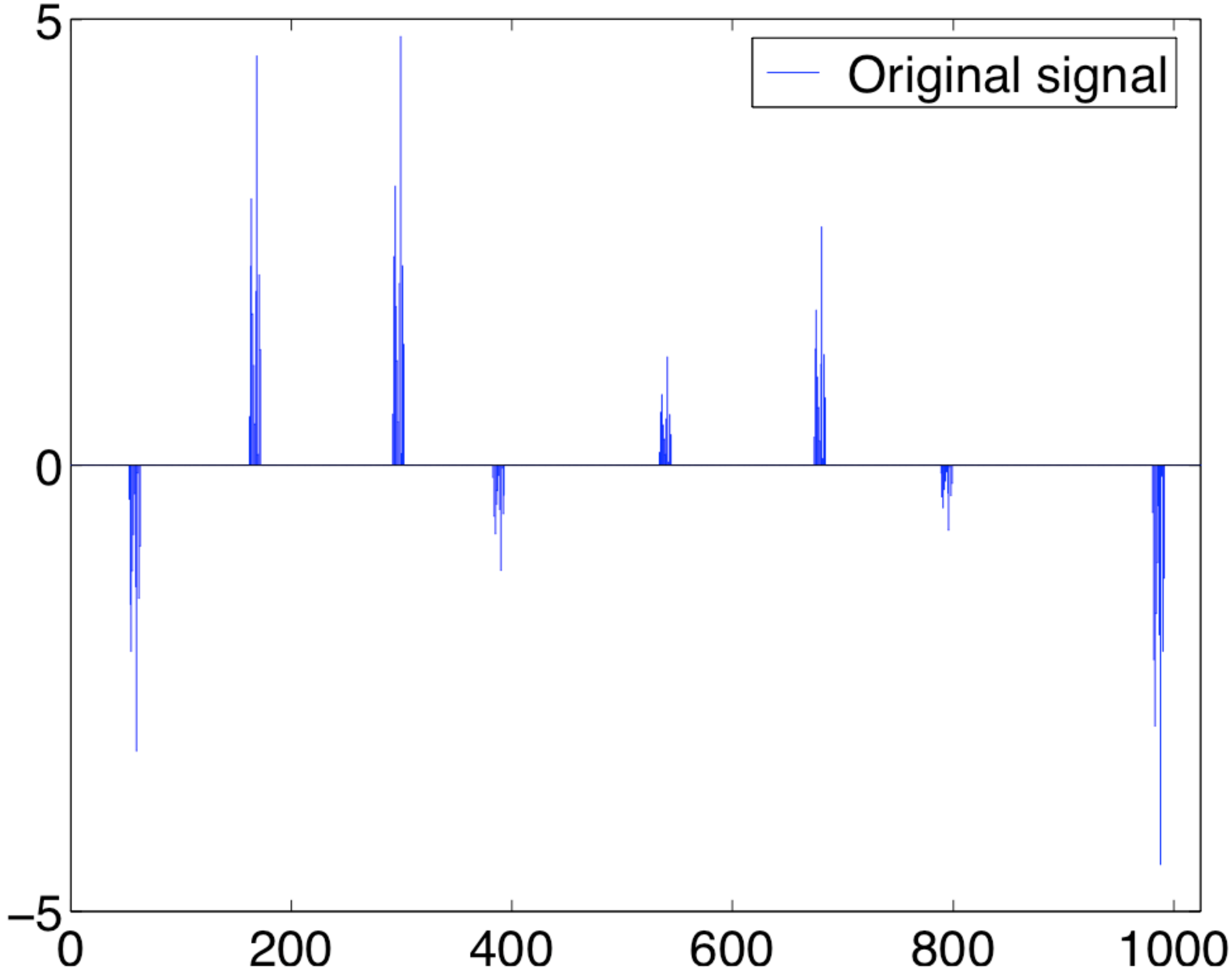
When things aren’t exactly sparse..

When things aren't exactly sparse..



- *S*-sparse signal *convolved* with an *F*-sparse impulse response

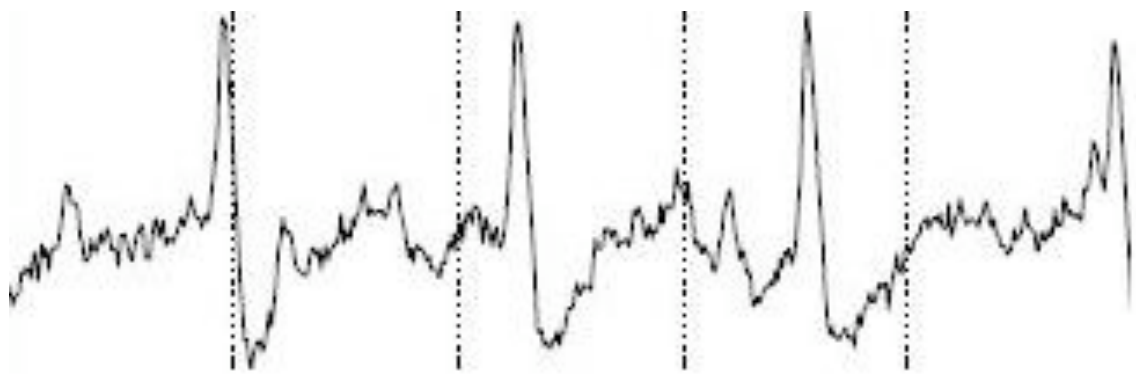
When things aren't exactly sparse..



$$z = Hx = Xh$$

Streams of pulses

- Neuronal spike trains



- UWB signals
- Astronomical imaging
- Etc.

Streams of pulses

- Overall sparsity: $K = SF$

- Model-based CS: $M = O(K + \log(m_K))$

- BUT #{degrees of freedom} = $O(S + F)$

Streams of pulses

- Overall sparsity: $K = SF$

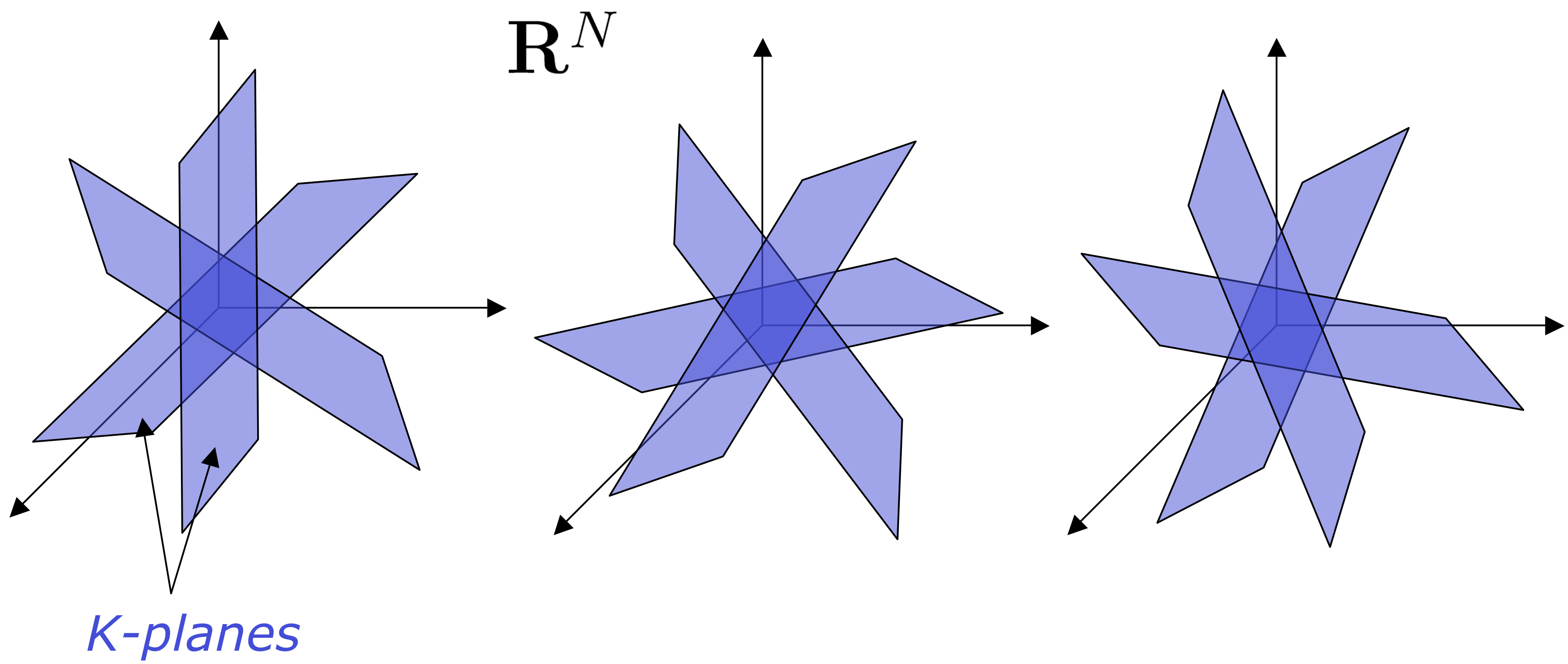
- Model-based CS: $M = O(K + \log(m_K))$

- BUT #{degrees of freedom} = $O(S + F)$

- **Can we do better?**

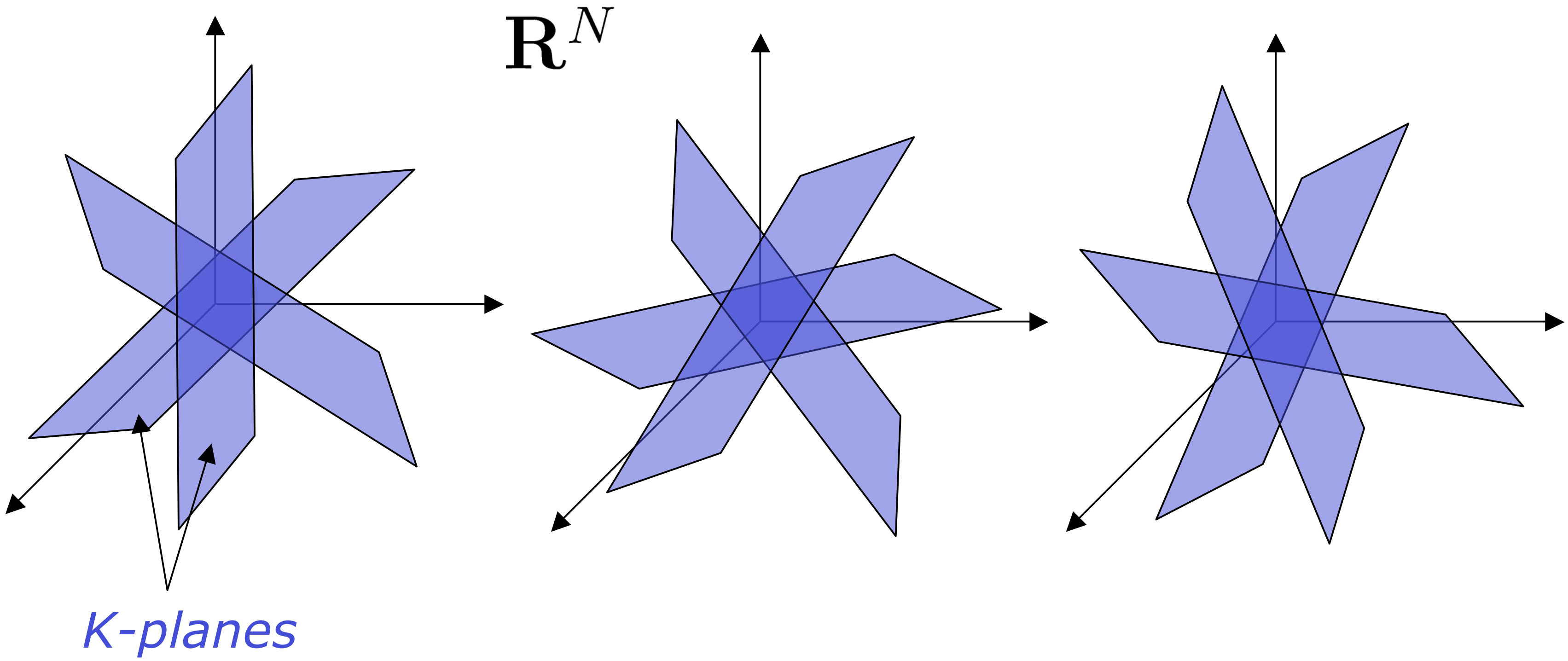
Geometry of signal set

- *Infinite* union of subspaces



Geometry of signal set

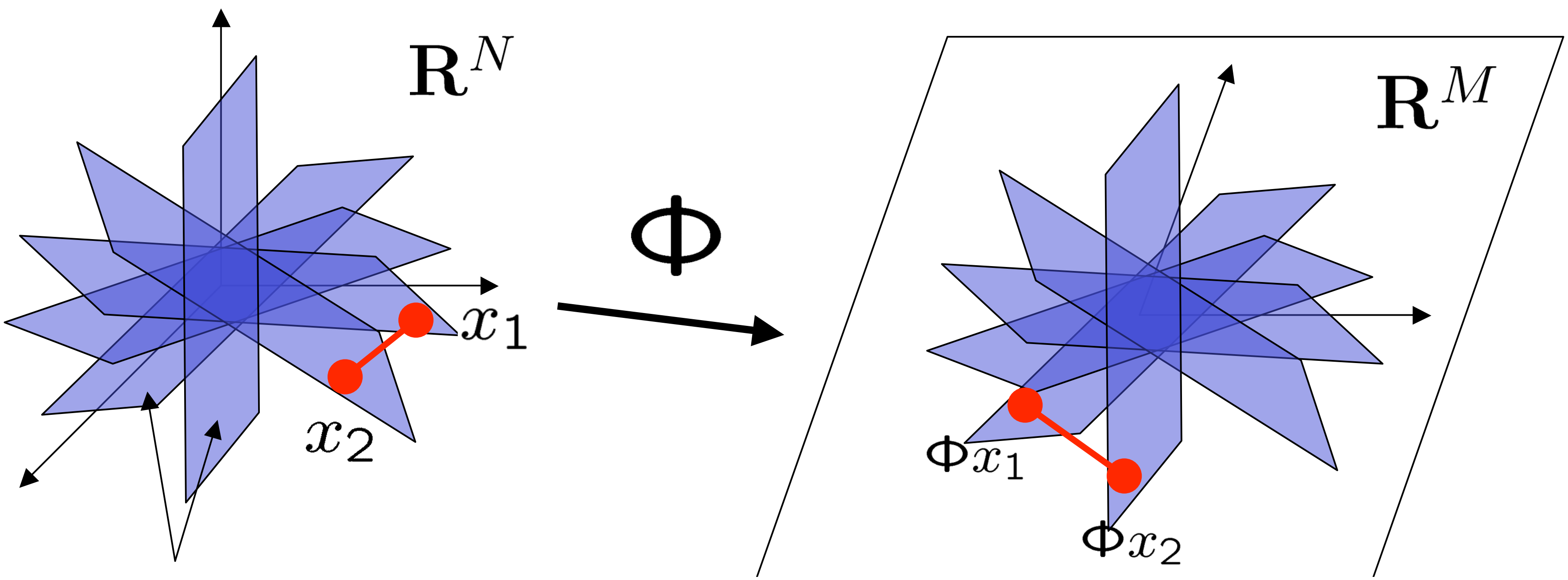
- *Infinite* union of subspaces



- Very small subset of the set of all SF-sparse signals

Sampling Bound

- **RIP for pulse streams**



Infinite union of subspaces

$$M = O(S + F + S \log(N/S))$$

Recovery

- Similar in spirit to **blind deconvolution**
- Slightly different goal: recover the *pulse stream*
- Iterate between estimating spikes and filter coefficients

Recovery algorithm

Algorithm 1

Inputs: Projection matrix Φ, measurements y, model parameters Δ, S, F.

Output: $\mathcal{M}_{S,F}^{\Delta}$-sparse approximation $\widehat{z}$ to true signal z

$\widehat{x} = 0$, $\widehat{h} = (\mathbf{1}_F^\top, 0, \ldots, 0)$; $i = 0$ {initialize}

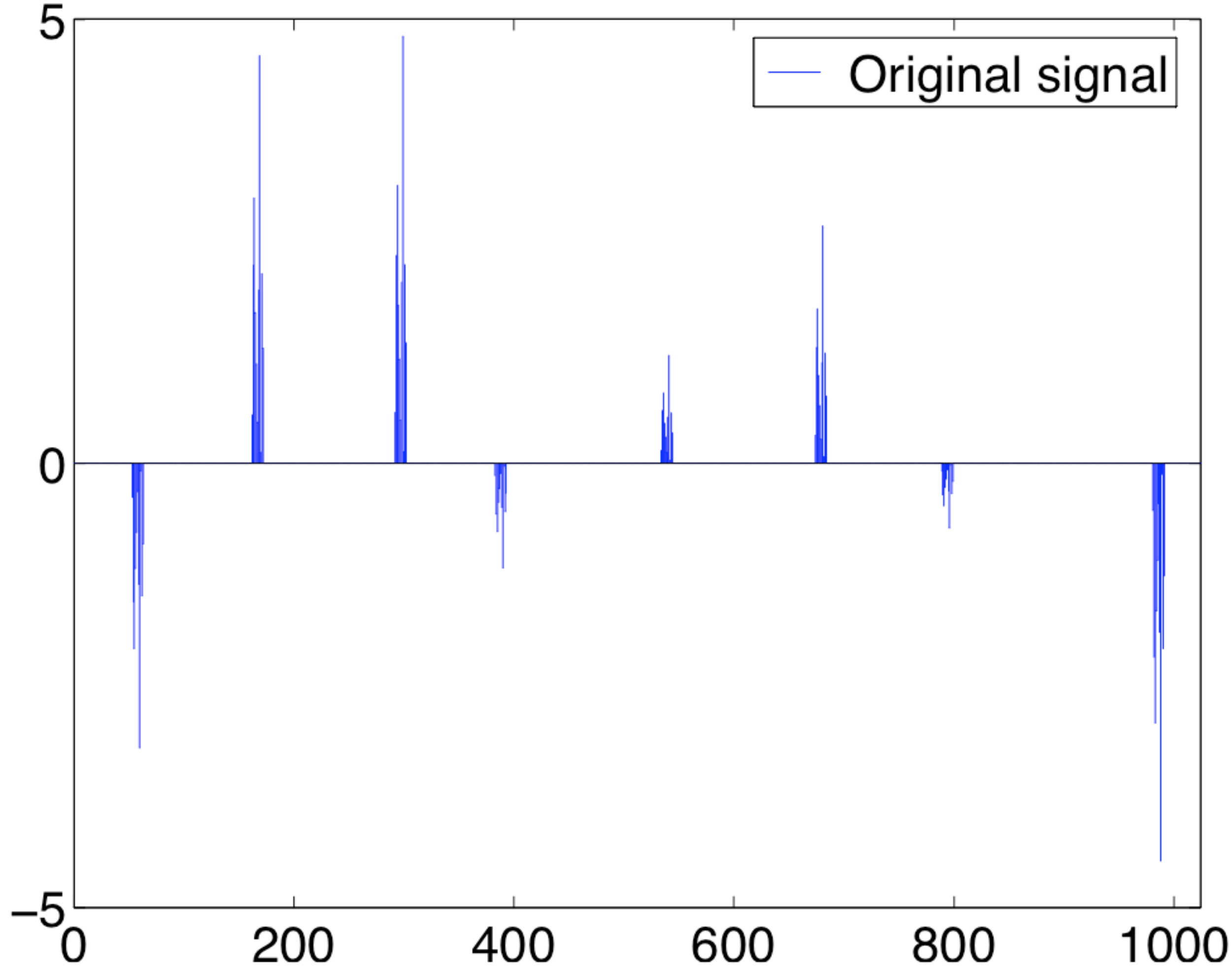
while halting criterion false **do**

1. $i \leftarrow i + 1$
2. $\widehat{z} \leftarrow \widehat{x} * \widehat{h}$ {current signal estimate }
3. $\widehat{H} = \mathbb{C}(\widehat{h}), \Phi_h = \Phi\widehat{H}$ {form dictionary for spike domain}
4. $e \leftarrow \Phi_h^T(y - \Phi_h\widehat{z})$ {residual}
5. $\Omega \leftarrow \mathrm{supp}(\mathbb{D}_2(e))$ {prune residual according to $(2S, 2\Delta)$ model}
6. $T \leftarrow \Omega \cup \mathrm{supp}(\widehat{x}_{i-1})$ {merge supports}
7. $b|_T \leftarrow (\Phi_h)_T^\dagger y$, $b|_{T^C} = 0$ {update spike estimate}
8. $\widehat{x} \leftarrow \mathbb{D}(b)$ {prune spike estimate according to (S, Δ) model}
9. $\widehat{X} = \mathbb{C}(\widehat{x}), \Phi_x = \Phi\widehat{X}$ {form dictionary for filter domain }
10. $\widehat{h} \leftarrow \Phi_x^\dagger y$ {update filter estimate }

end while

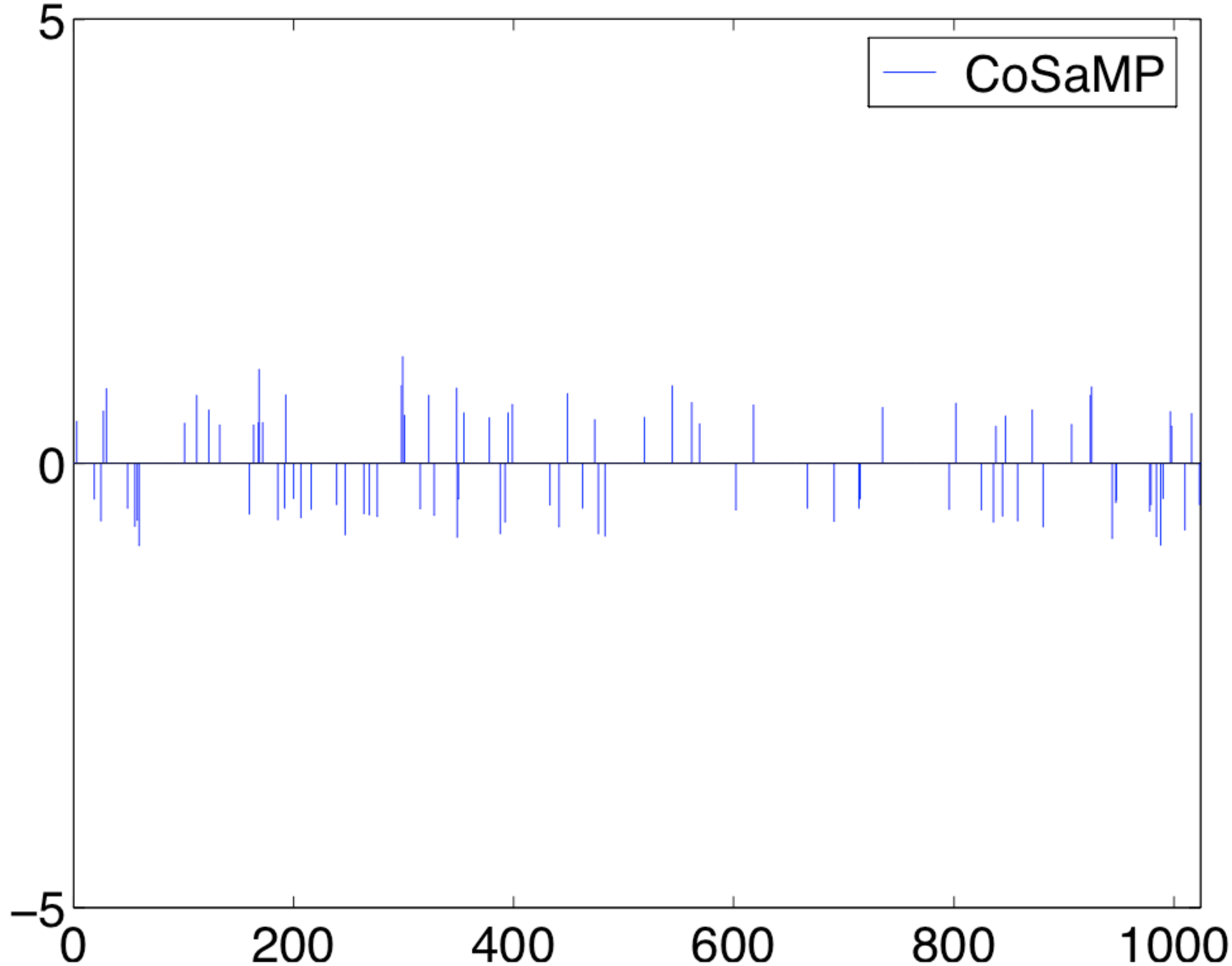
return $\widehat{z} \leftarrow \widehat{x} * \widehat{h}$

Numerical example



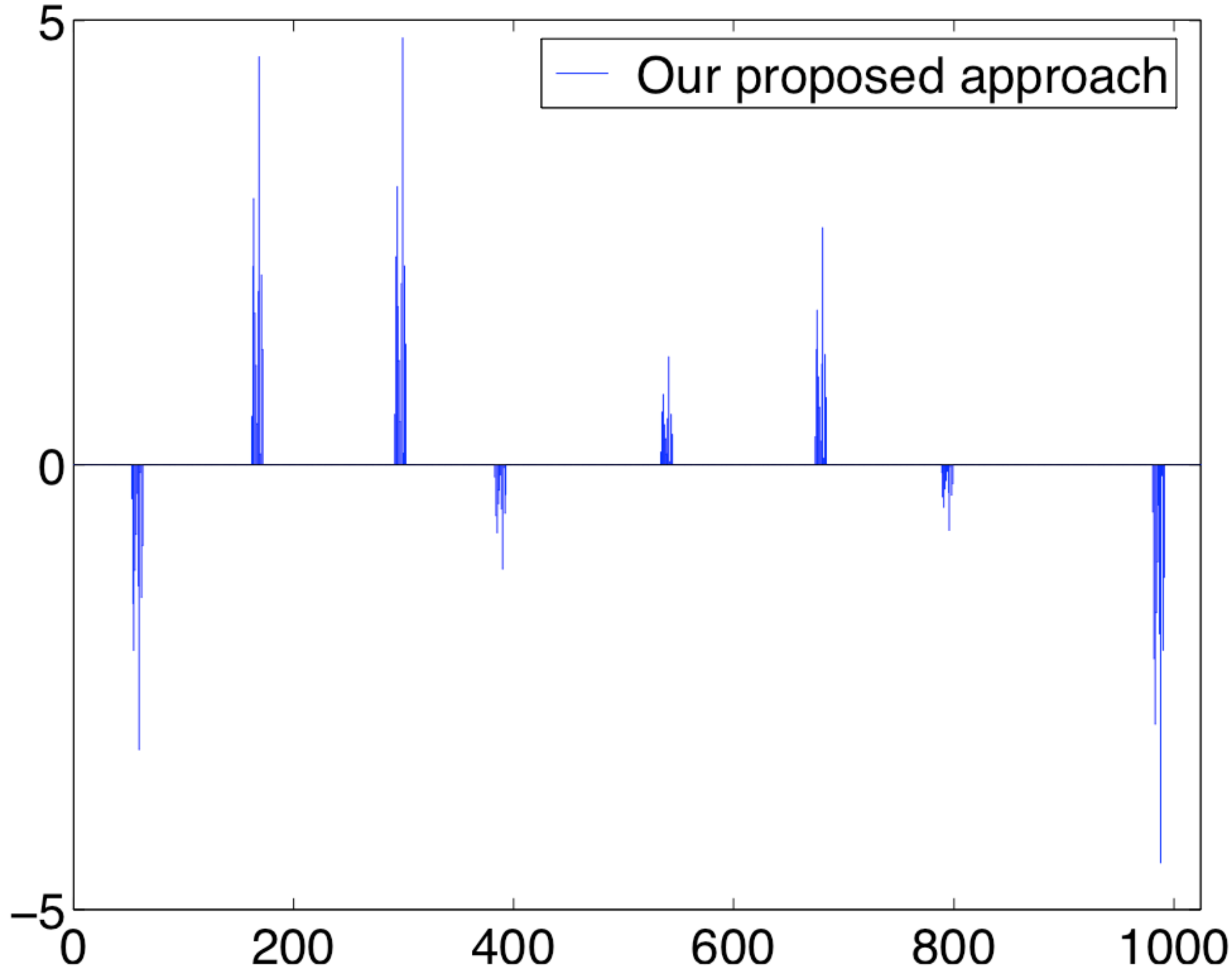
Stream of pulses: N = 1024, S = 8, F = 11

Numerical example



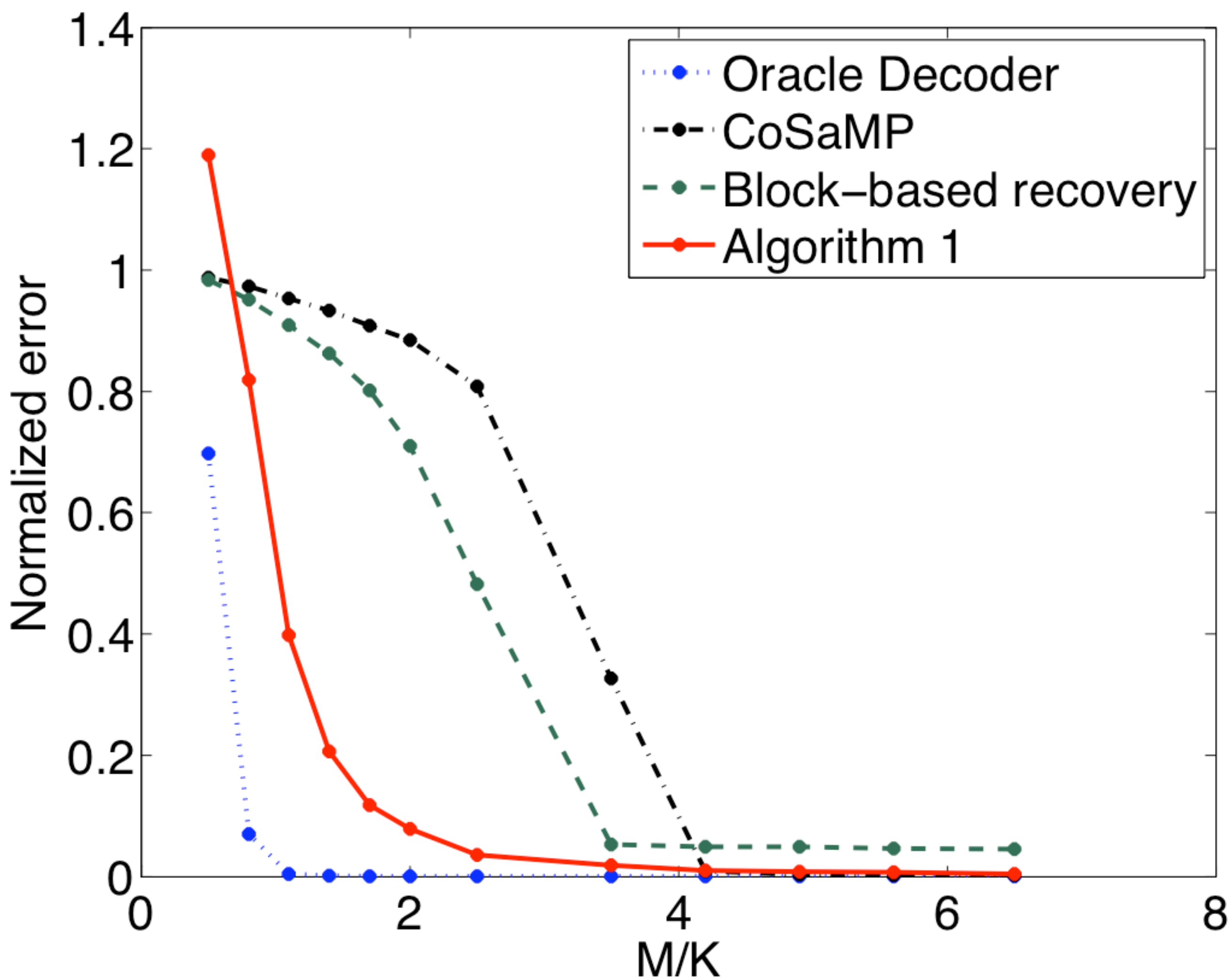
CS recovery using M = 90 measurements

Numerical example

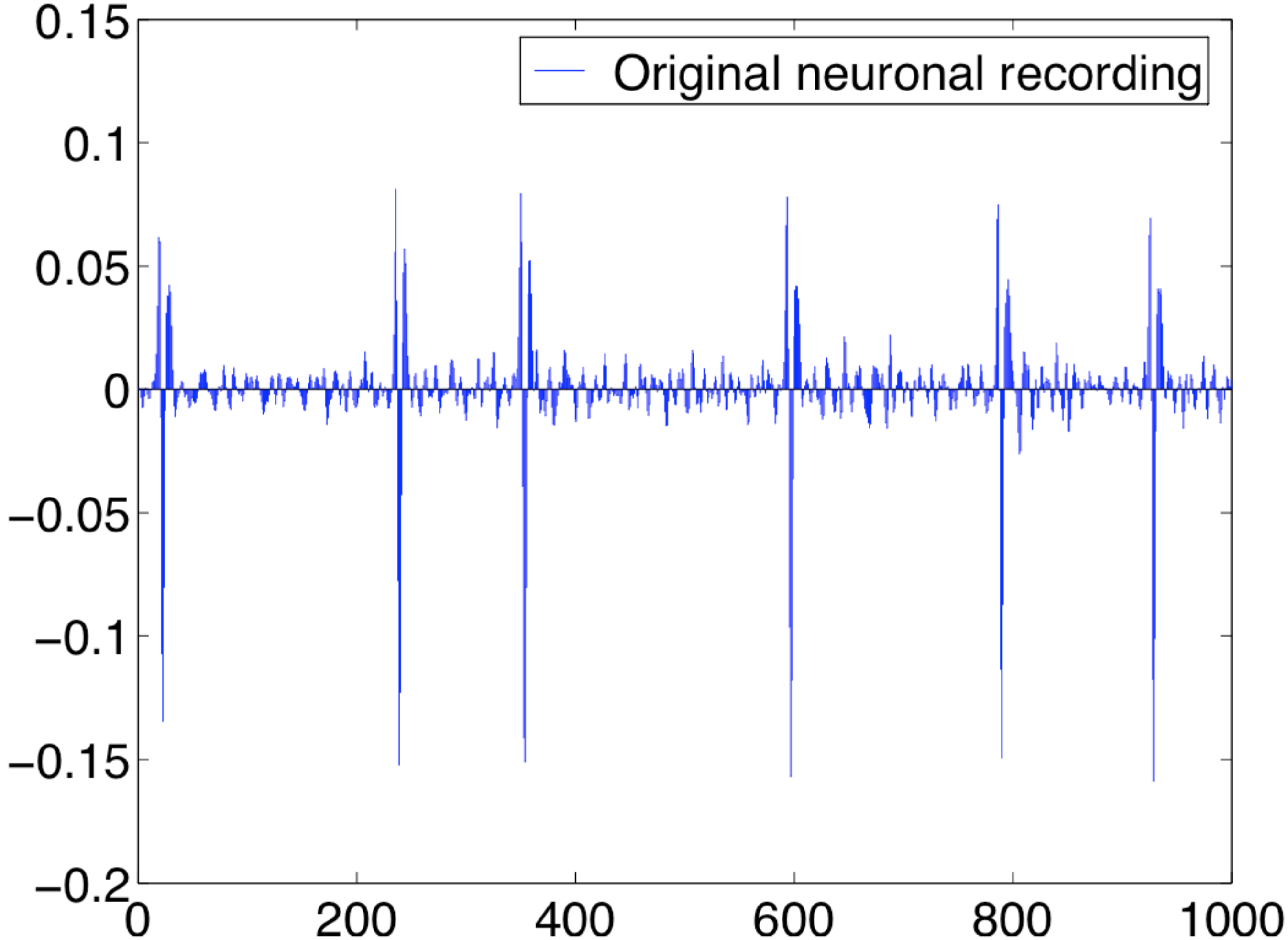


CS recovery using M = 90 measurements

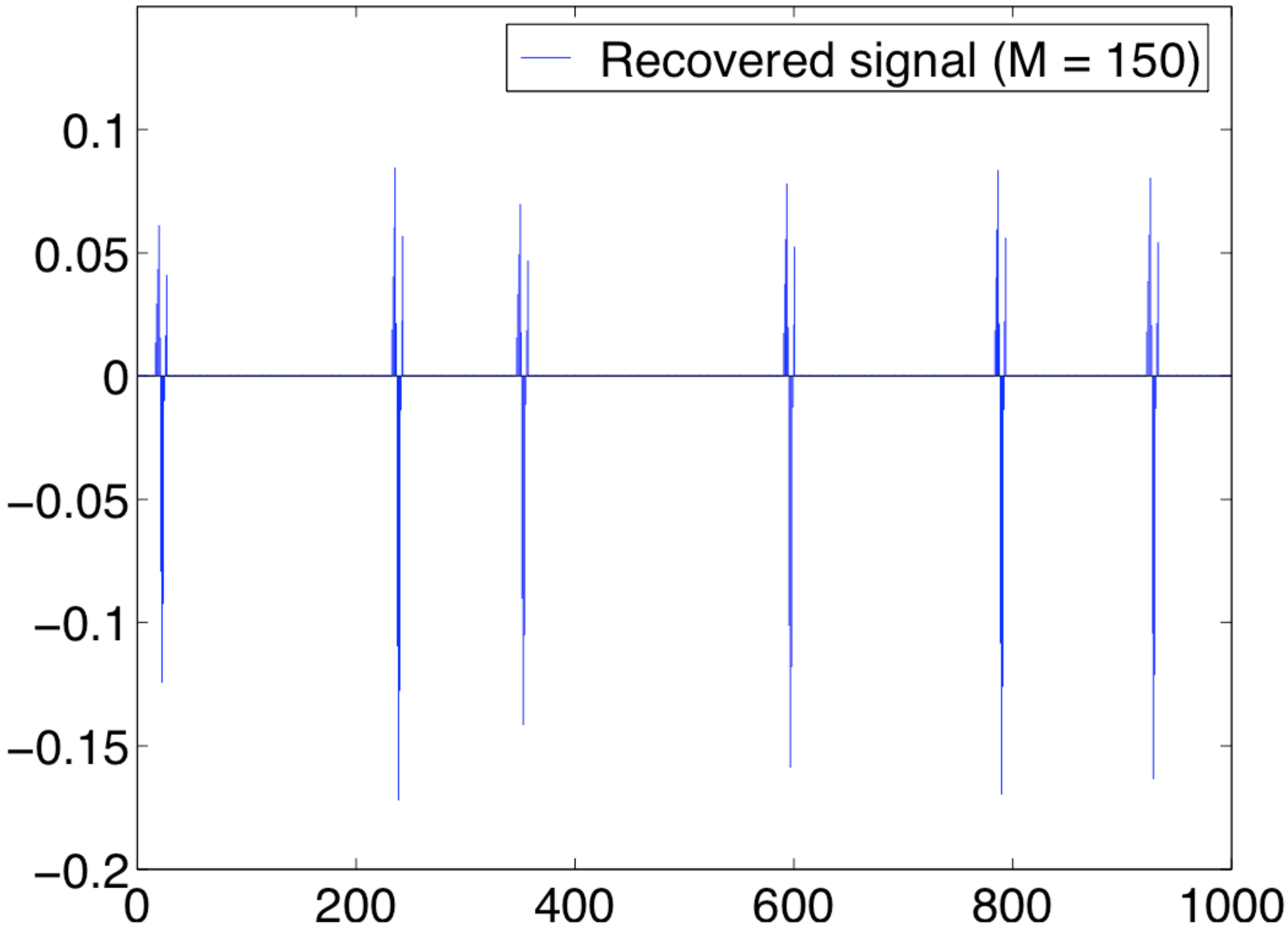
Monte Carlo Simulation



Real-data experiment

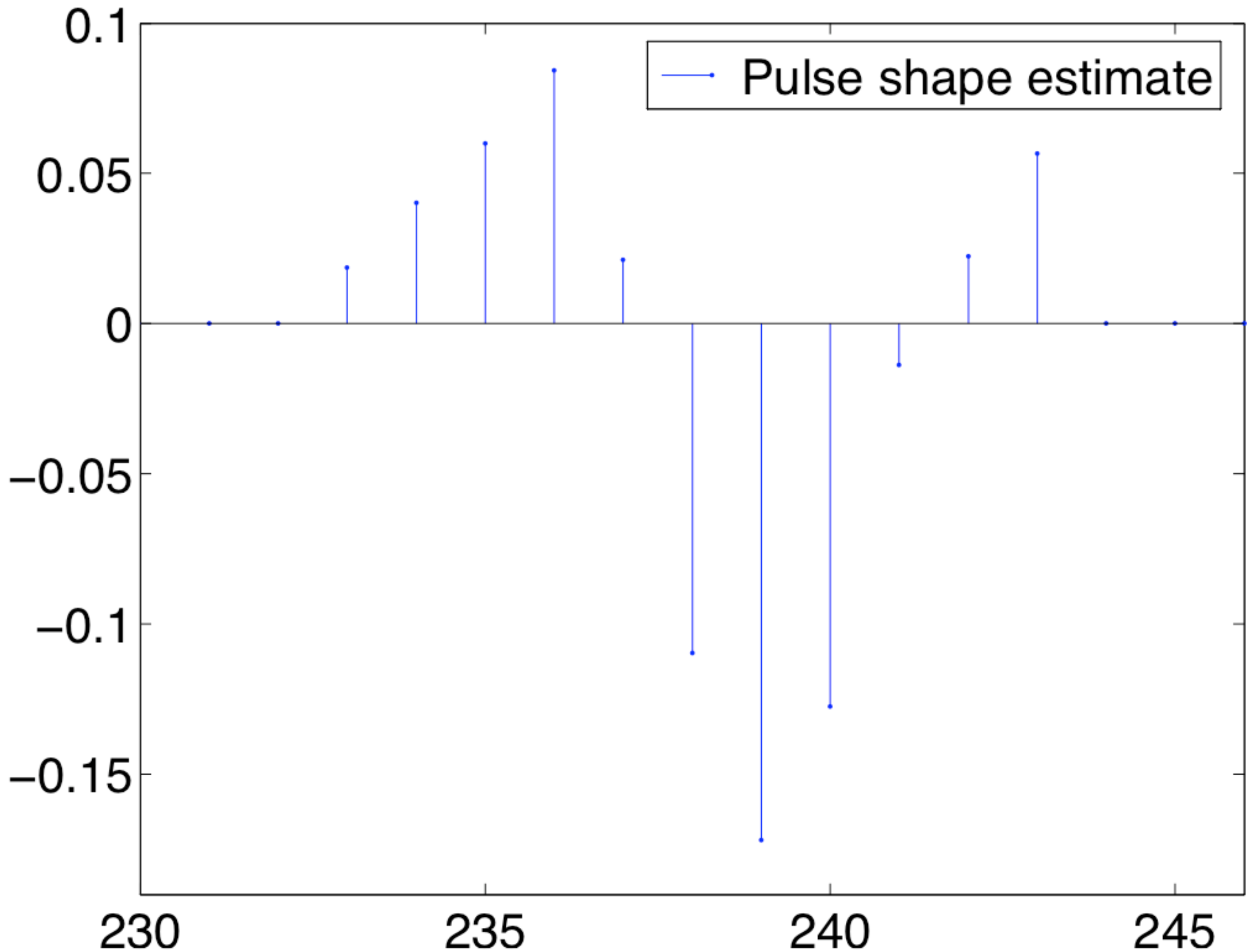


Real-data experiment



CS recovery using the pulse-stream model

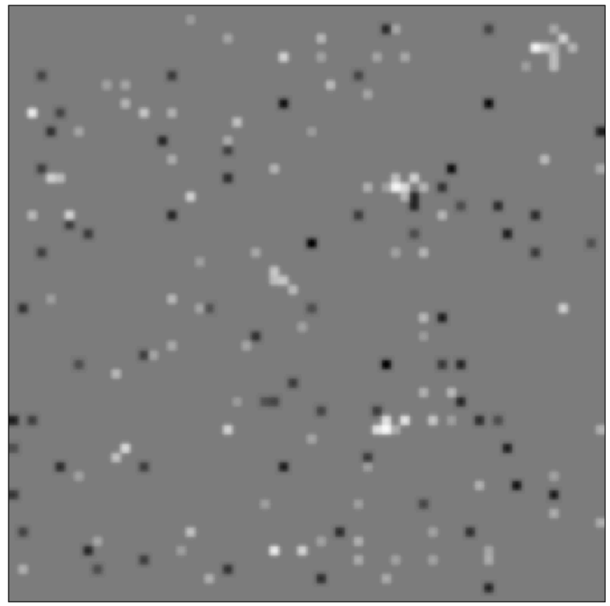
Real data experiment



Extension to 2D

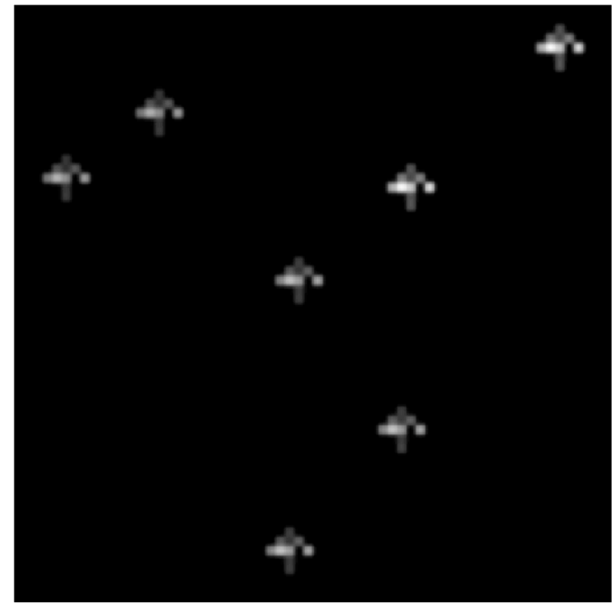
Original image

(N = 4096, K = 175)



CoSaMP (M = 290)

MSE = 16.95



Pulse-field (M = 290)

MSE = 0.07

Summary

- Why CS works: stable embedding for signals with concise geometric structure

- Contribution: a CS framework for pulse streams

  Advantages: provably fewer measurements
  simple, flexible algorithm

www.dsp.rice.edu/cs

- Blank slide

- Another blank slide

- Yet another blank slide