Large-Scale Convex Optimization for Sparse Recovery

Justin Romberg, California Institute of Technology

Collaborators: *Emmanuel Candès*, *Michael Slomka*

Overview

- Compressive sampling/imaging: recover a sparse signal $x_0 \in \mathbb{R}^N$ from $K \ll N$ incoherent measurements $y = \Phi x_0$
- Recovery process consists of solving certain ℓ_1 and total-variation minimization problems
- This talk: Algorithms for solving these optimization programs
- Programs fall into two classes:
  - linear programs (LPs)
  - second-order cone programs (SOCPs)
- Tremendous progress in the last decade in solving problems of this type
- We have implemented simple (but very effective) solvers, code available at `www.l1-magic.org`

ℓ_1 minimization

- ℓ_1 with equality constraints ("Basis Pursuit") can be recast as a linear program (Chen, Donoho, Saunders 1995 and others)

$$\min_{x} \; \|x\|_{\ell_1} \quad \text{subject to} \quad \Phi x = y$$

$$\Updownarrow$$

$$\min_{u,x} \; \sum_t u(t) \quad \text{subject to} \quad \begin{aligned} -u \;\le\; x \;\le\; u \\ \Phi x = \; y \end{aligned}$$

Total-Variation Minimization

- The Total Variation functional is a "sum of norms"

$$\begin{aligned}\mathbf{TV}(x) &= \sum_{i,j} \sqrt{(x_{i+1,j} - x_{i,j})^2 + (x_{i,j+1} - x_{i,j})^2} \\ &= \sum_{i,j} \|D_{i,j}x\|_2 \qquad D_{i,j}x = \begin{bmatrix} x_{i+1,j} - x_{i,j} \\ x_{i,j+1} - x_{i,j} \end{bmatrix}\end{aligned}$$

- Total variation minimization can be written as a second-order cone program (Boyd et. al, 1997, and others)

$$\min_x \mathbf{TV}(x) := \sum_{i,j} \|D_{i,j}x\|_2 \quad \text{subject to} \quad \|\Phi x - y\|_2 \le \epsilon$$

$$\Updownarrow$$

$$\min_{u,x} \sum_{i,j} u_{i,j} \quad \text{subject to} \quad \begin{aligned} \|D_{i,j}x\|_2 &\le u_{i,j}, \quad \forall i,j \\ \|\Phi x - y\|_2 &\le \epsilon \end{aligned}$$

Other examples

- ℓ_1 minimization with complex coefficients (sum of norms)

$$\min_x \ \|x\|_{\ell_1} := \min_x \ \sum_t \sqrt{\mathrm{Real}(x(t))^2 + \mathrm{Imag}(x(t))^2}$$

$$\text{s.t.} \quad Ax = b$$

$\Rightarrow$ SOCP

- Dantzig Selector

$$\min_x \ \|x\|_{\ell_1} \quad \text{subject to} \quad \|\Phi^*(\Phi x - y)\|_\infty \leq \epsilon$$

$$\Updownarrow$$

$$\min_{u,x} \ \sum_t u(t) \quad \text{subject to} \quad \begin{array}{rcl} -u & \leq \ x \ \leq & u \\ -\epsilon \cdot \mathbf{1} & \leq \Phi^*(\Phi x - y) \leq & \epsilon \cdot \mathbf{1} \end{array}$$

$\Rightarrow$ LP

Primal-Dual Algorithms for LP

- Standard LP:

$$\min_{\boldsymbol{x}} \ \langle \boldsymbol{c}, \boldsymbol{x} \rangle \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \quad \boldsymbol{x} \leq \boldsymbol{0}$$

- Karush-Kuhn-Tucker (KKT) conditions for optimality:
  Find $\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{\nu}^\star$ such that

$$\begin{array}{ccc} \boldsymbol{A}\boldsymbol{x}^\star = \boldsymbol{b} & \boldsymbol{c} + \boldsymbol{A}^*\boldsymbol{\nu}^\star + \boldsymbol{\lambda}^\star = \boldsymbol{0} & \boldsymbol{x}_i^\star \boldsymbol{\lambda}_i^\star = \boldsymbol{0}, \quad \forall \boldsymbol{i} \\ \boldsymbol{x}^\star \leq \boldsymbol{0} & \boldsymbol{\lambda}^\star \geq \boldsymbol{0} & \end{array}$$

- Primal-dual algorithm:
  - Relax: use $\boldsymbol{x}_i \boldsymbol{\lambda}_i = \boldsymbol{1}/\boldsymbol{\tau}$, increasing $\boldsymbol{\tau}$ at each iteration
  - Linearize system

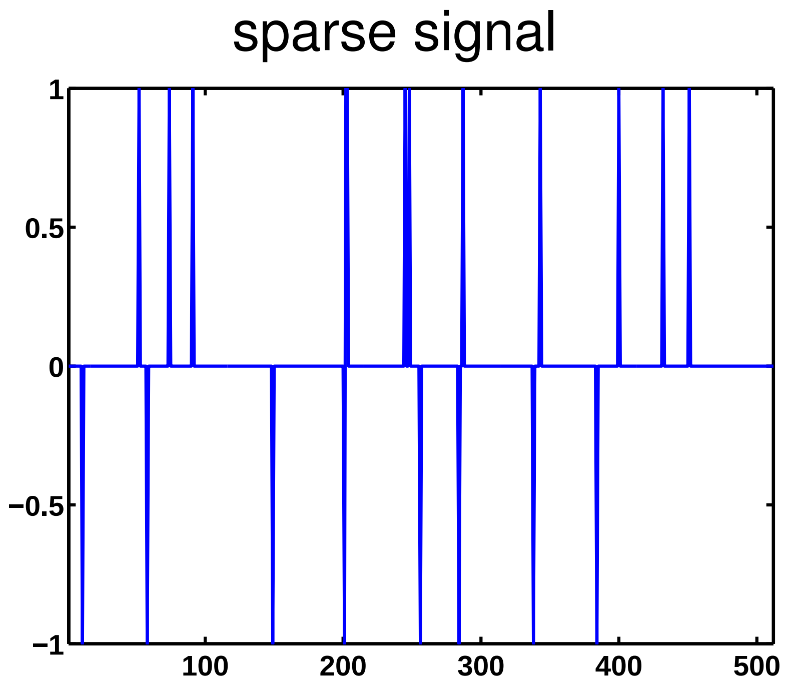
$$\begin{pmatrix} \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \\ \boldsymbol{c} + \boldsymbol{A}^*\boldsymbol{\nu} + \boldsymbol{\lambda} \\ \boldsymbol{x}_i \boldsymbol{\lambda}_i \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{1}/\boldsymbol{\tau} \end{pmatrix}$$

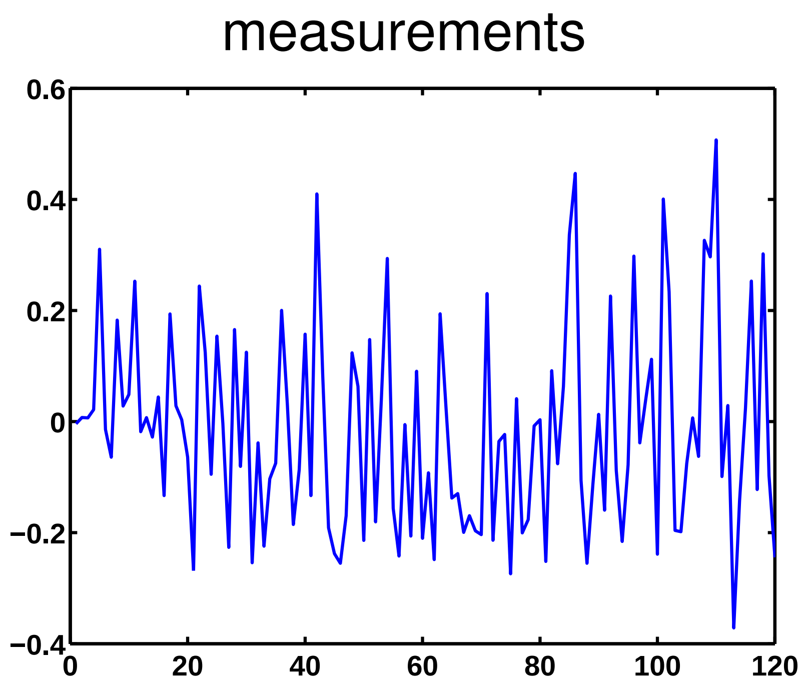
  - Solve for step direction, adjust length to stay in interior ($\boldsymbol{x} \leq \boldsymbol{0}, \ \boldsymbol{\lambda} \geq \boldsymbol{0}$)

Newton Iterations

- Newton: solve $f(x) = 0$ iteratively by solving a series of linear problems
  - At x_k,
$$f(x_k + \Delta x_k) \approx f(x_k) + \Delta x_k f'(x_k)$$
  - Solve for Δx_k such that $f(x_k) + \Delta x_k f'(x_k) = 0$
  - Set $x_{k+1} = x_k + \Delta x_k$
  - Repeat
- Each Newton iteration requires solving a linear system of equations
- Bottleneck of the entire procedure:
  We need to solve a series of $K \times K$ systems ($K =$ # constraints)
- Each step is expensive, but we do not need many steps
  - Theory: need $O(\sqrt{N})$ steps
  - Practice: need $\approx$ 10–40 steps

Example





- $N = 512$, $K = 120$
- Recover using ℓ_1 minimization with equality constraints
- Requires 12 iterations to get within 10^{-4} (4 digits)
- Takes about 0.33 seconds on high-end desktop Mac (Matlab code)

Large-Scale Systems of Equations

- The system we need to solve looks like

$$\boldsymbol{A\Sigma A^* \Delta x = w}$$

  $\boldsymbol{A : K \times N}$
  $\boldsymbol{\Sigma : N \times N}$ diagonal matrix; changes at each iteration

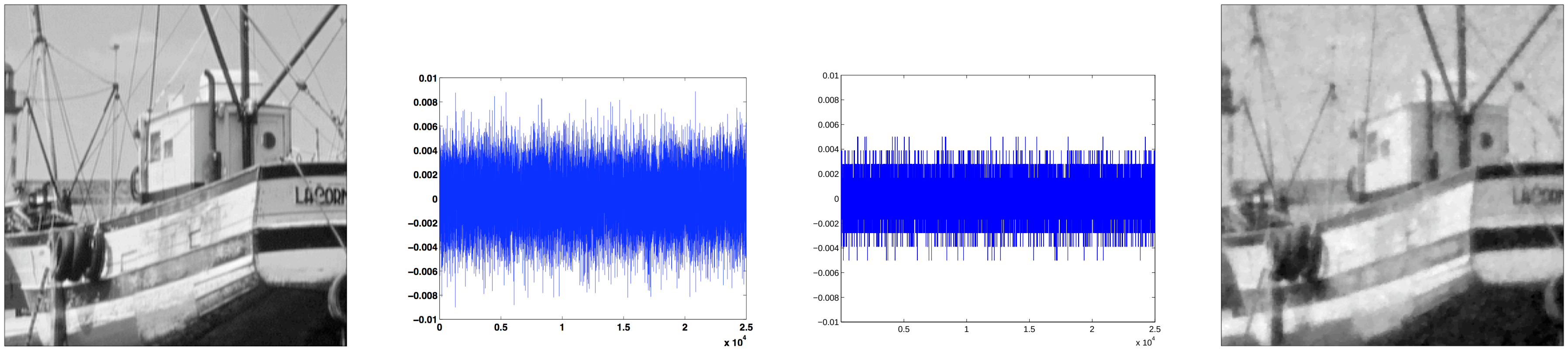
- Computation: $\boldsymbol{O(NK^2)}$ to construct, $\boldsymbol{O(K^3)}$ to solve
- Large scale: we must use implicit algorithms (e.g. Conjugate Gradients)
  - iterative
  - requires an application of $\boldsymbol{A}$ and $\boldsymbol{A^*}$ at each iteration
  - number of iterations depends on condition number
- $\boldsymbol{A = \Phi\Psi^*}$
  $\boldsymbol{\Phi = K \times N}$ measurement matrix
  $\boldsymbol{\Psi = N \times N}$ sparsity basis
- For large-scale Compressive Sampling to be feasible,
  we must be able to apply $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ (and $\boldsymbol{\Phi^*}$, $\boldsymbol{\Psi^*}$) quickly
  ($\boldsymbol{O(N)}$ or $\boldsymbol{O(N \log N)}$)

Fast Measurements

- Say we want to take $\mathbf{20,000}$ measurements of a $\mathbf{512 \times 512}$ image ($\boldsymbol{N = 262,144}$)
- If $\boldsymbol{\Phi}$ is Gaussian, with each entry a float, it would take more than an entire DVD just to hold $\boldsymbol{\Phi}$
- Need fast, implicit, noise-like measurement systems to make recovery feasible
- Partial Fourier ensemble is $\boldsymbol{O(N \log N)}$ (FFT and subsample)
- Tomography: many fast unequispaced Fourier transforms, Dutt and Rohklin, Pseudopolar FFT of Averbuch et. al
- Noiselet system of Coifman and Meyer
  - perfectly incoherent with Haar system
  - performs the same as Gaussian (in numerical experiments) for recovering spikes and sparse Haar signals
  - $\boldsymbol{O(N)}$

Large Scale Example

image measure quantize recover



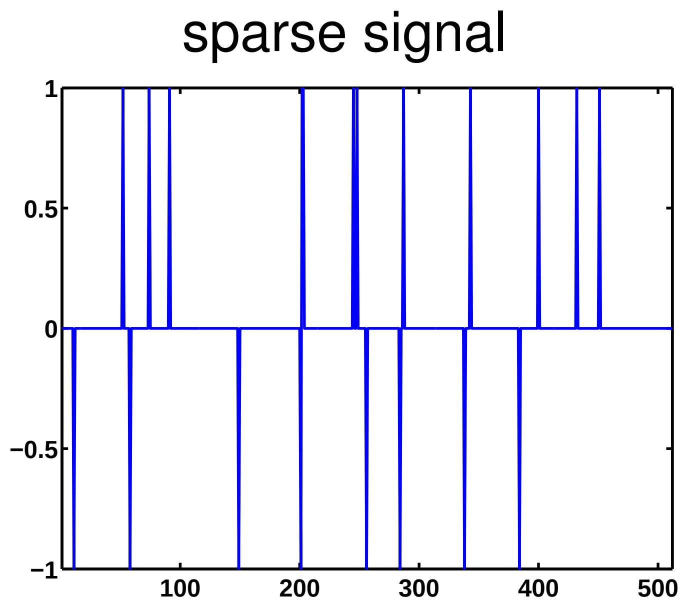
- $N = 256 \times 256$, $K = 25000$
- Measure using "scrambled Fourier ensemble"
  (randomly permute the columns of the FFT)
- Recover using $\mathbf{TV}$-minimization with relaxed constraints
- Recovery takes ≈ 5 minutes on high-end desktop Mac
- Vanilla log barrier SOCP solver (in Matlab)
- Note: Noise and approximate sparsity help us here

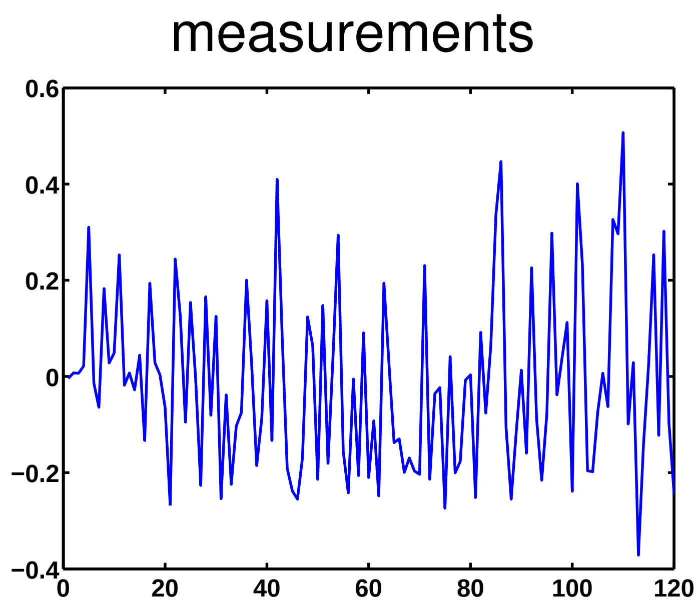
Conditioning

$$\boldsymbol{A\Sigma A^* \Delta x = w}$$

- When recovering a truly sparse signal, the system above becomes very ill-conditioned as we approach the solution
- $\boldsymbol{\sigma} = \mathbf{diag}(\boldsymbol{\Sigma})$
  $\boldsymbol{\sigma(t) \longrightarrow 1}$, for $\boldsymbol{t} \in \mathbf{supp}\, \boldsymbol{x_0}$
  $\boldsymbol{\sigma(t) \longrightarrow 0}$, for $\boldsymbol{t} \notin \mathbf{supp}\, \boldsymbol{x_0}$
  $\Rightarrow \boldsymbol{A\Sigma A^*}$ becomes low-rank
- Small scale: even with exact inversion, we can only get within 4 or 5 digits
- Large scale: many CG iterations are needed to find a good step direction
- Common problem in interior point methods for LP

Conditioning Example





Iteration	PDGap	Cond #
7	2.9e0	1.5e4
8	3.2e-1	6.5e5
9	3.5e-2	2.1e6
10	3.8e-3	1.4e8
11	4.2e-4	1.2e10
12	4.5e-5	9.6e11

Basis Pursuit as Decoding

- It is possible to reformulate BP in such a way that this conditioning problem disappears

$$\begin{array}{ll} \min_{x} \ \|x\|_{\ell_1} & \Leftrightarrow \qquad \min_{h} \ \|Qh + x_0\|_{\ell_1} \\ \text{s.t.} \quad Ax = b & \end{array}$$

- Columns of Q span the nullspace of A: $AQ = 0$
  $Q : N \times (N - K)$
- x_0 is any feasible point
- Eliminate equality constraints by restricting search to nullspace
- If A = Fourier transform on Ω, Q^* = Fourier transform on Ω^c
- Same form as "Decoding by Linear Programming" (Candès and Tao, 2004)

Conditioning of ℓ_1 Approximation

$$Q^* \Sigma Q \Delta h = w$$

- We know have a $(N-K) \times (N-K)$ system instead of $K \times K$ (inconsequential for things like partial Fourier ensembles)

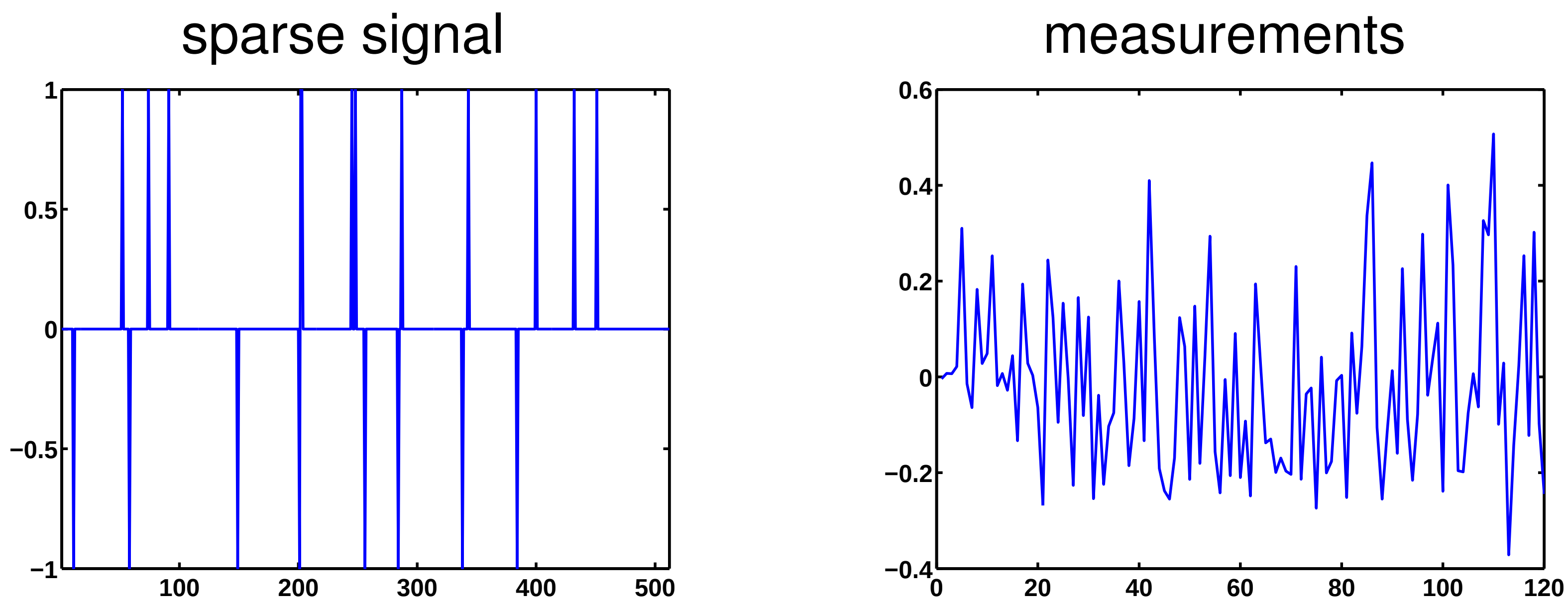
- $\sigma = \mathrm{diag}(\Sigma)$
  $\sigma(t) \to 1$, for $t \notin \mathrm{supp}\, x_0$
  $\sigma(t) \to 0$, for $t \in \mathrm{supp}\, x_0$
  $\Rightarrow\ Q^* \Sigma Q$ remains full rank

- In fact, the Uniform Uncertainty Principle implies that close to the solution

$$\mathrm{cond}(Q^* \Sigma Q) \sim \frac{N}{K}$$

- All of the conditioning problems disappear

Conditioning Example, II



Recovery via the ℓ_1-approximation reformulation

Iteration	PDGap	Cond #
7	5.3e-1	129
8	5.8e-2	73
9	6.3e-3	71
10	6.8e-4	70
11	7.5e-5	70

Large Scale Example

- $N = 1024^2 \approx 10^6$, $K = 96,000$
- Perfectly sparse image (in wavelet domain), $S = 25,000$
- Recovered to 4 digits in 50 iterations
  (5 digits in 52 iterations, 6 digits in 54 iterations,$\ldots$)
- Recovery time was less than 40 minutes on high-end desktop Mac

Summary

- Compressive sampling recovery programs (ℓ_1 and TV minimization) can be recast as linear programs or second-order cone programs
- Efficiently implemented using standard interior point methods
- For sparse recovery, removing the equality constraints makes the procedure incredibly well-conditioned numerically
- Recovering megapixel images is computationally feasible
- Current work:
  - Showing that the Newton system is well-conditioned everywhere
  - Similar conditioning techniques for the relaxed problems
  - More sophisticated SOCP solvers (e.g. a primal-dual algorithm similar to LP)
  - More sophisticated SOCP models for images (cleaner recovery in the noisy/non-sparse cases)
- Code at `www.l1-magic.org`