Compressed Sensing in Python

Sercan Yıldız
syildiz@samsi.info

February 27, 2017

Outline

A Brief Introduction to Compressed Sensing

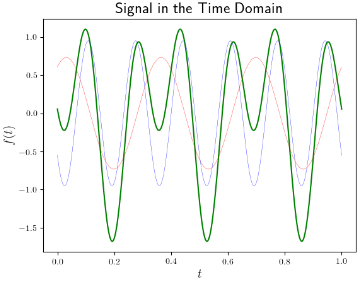
A Briefer Introduction to CVXOPT

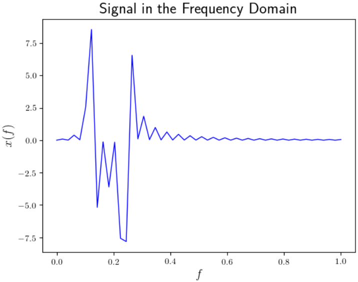
Examples

A Brief Introduction to Compressed Sensing

Signal Processing and Compression

- Signals over time and/or space
- Often, signals are sparse in an appropriate domain.





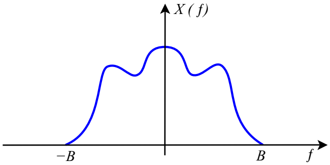
- Basic idea behind lossy sound/image compression:
  - Transform signal to the frequency domain.
  - Keep frequencies with the largest magnitudes, discard the rest.
  - Examples: MP3, JPEG, MPEG

Sampling

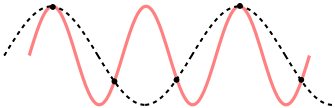
- Sampling at discrete points:
  - Reduces a continuous-time (analog) signal to a discrete-time (digital) signal
  - If the signal is bandlimited (bounded frequency) and the sampling rate is large enough, the signal can be exactly recovered:

Theorem (Nyquist-Shannon sampling theorem)

A bandlimited signal can be exactly reconstructed from its samples if the sampling rate is more than twice the maximum frequency in the signal.



A bandlimited signal in the frequency domain. Image source: https://upload.wikimedia.org/wikipedia/commons/f/f7/Bandlimited.svg


Two sine waves can have identical samples if the sampling rate is not large enough. Image source: https://upload.wikimedia.org/wikipedia/commons/a/af/CPT-sound-nyquist-thereom-1.5percycle.svg

- Compressed sensing combines compression and sampling with efficient sampling protocols that capture and condense the information content in a sparse signal into a small amount of data.

Compressed Sensing (CS)

- Perfect reconstruction can be possible at sub-Nyquist sampling rates if additional information about the signal (such as sparsity) is available.
- Why are low sampling rates attractive?
  - Number of sensors may be limited.
  - Measurements may be expensive.
  - Sensing process may be slow.
- Reconstruction of undersampled signal requires optimization!
- In the remainder, we consider discrete signals of finite length...and make things a little more concrete.

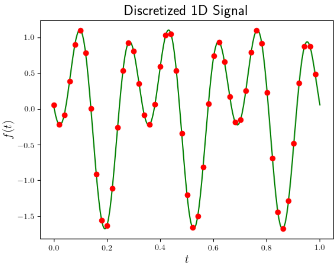


Image source: https://issc.science.lsst.org/sites/default/files/SAMSIBldg1-web.jpg

Compressed Sensing (CS)

- Given: Signal f with the sparse representation $f = \Psi x$ for some square unitary matrix Ψ
- We would like to design an $n \times m$ sensing matrix $\hat{\Phi}$ (for $m << n$) that captures as much information about f as possible. The matrix $\hat{\Phi}$ will generate the observations

$$y = \hat{\Phi}^\top f.$$

- Questions:
  - How do we design $\hat{\Phi}$?
  - How do we reconstruct f from y?

Sensing Matrices

- A key property of good sensing matrices is their incoherence with Ψ.
- The *coherence* of two orthogonal matrices Φ and Ψ is

$$\mu(\Phi, \Psi) = \sqrt{n} \max_{j,k} \Phi_j^\top \Psi_k.$$

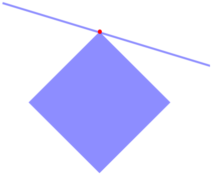
- Coherence measures the largest correlation between any two elements of Φ and Ψ.
- Examples of low coherence pairs:
  - Φ : identity matrix, Ψ : Fourier matrix
  - Φ : random orthogonal matrix, Ψ : fixed orthogonal matrix (whp)
- Given a low coherence pair (Φ, Ψ), we choose the sensing matrix $\hat{\Phi}$ as an $n \times m$ column submatrix of Φ.

Reconstructing the Signal

- Let $A = \hat{\Phi}^\top \Psi$.
- To recover the signal x in the frequency domain, we solve the "basis pursuit" problem

$$\min_x \|x\|_1 \quad \text{subject to} \quad Ax = y.$$

- $\ell 1$-norm promotes sparsity.



- How can we attack basis pursuit?
  - Reformulate as a linear program:

$$\min_{x,u} \sum_j u_j \quad \text{subject to} \quad Ax = y, \quad -u \leq x \leq u.$$

  - Reformulate as an $\ell 1$-regularized least squares problem:

$$\min_x \|Ax - b\|_2^2 + \lambda \|x\|_1.$$

  - Special-purpose algorithms

When does CS work?

Theorem (Candés and Romberg, 2007)

Suppose the true signal $\bar{x}$ is s-sparse. Let $\hat{\Phi}$ consist of m columns of Φ chosen uniformly at random. If

$$m \geq C\mu(\Phi, \Psi)s \log n$$

for some constant $C > 0$, then the basis pursuit problem recovers $\bar{x}$ with high probability.

- Taking $m \geq 4s$ seems to work well in practice.
- Signals are not always exactly sparse: Many coefficients of the true signal $\bar{x}$ may be small but not zero.

When does CS work?

Definition (Candés and Tao, 2005)

For each integer $s = 1, 2, \ldots$, the *isometry constant* δ_s of a matrix A is the smallest number such that

$$(1 - \delta_s)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta_s)\|x\|_2^2$$

holds for all s-sparse vectors x. We will say that A has the *restricted isometry property (RIP)* of order s if δ_s is "small."

- How is RIP useful?
  - If $\delta_{2s} \approx 0$, then $\|A(x_1 - x_2)\|_2^2 \approx \|x_1 - x_2\|_2^2$ for all s-sparse vectors x_1, x_2.
  - In other words, x_1 and x_2 remain distinguishable even after left-multiplication with A.

When does CS work?

Theorem (Candés, Romberg, and Tao, 2006)

Assume $\delta_{2s} < \sqrt{2} - 1$*. Let* $\bar{x}$ *be the true signal, and let* $\bar{x}_s$ *be the s-sparse vector consisting of the s largest (in absolute value) entries of* $\bar{x}$*. Then the solution* x^* *to the basis pursuit problem satisfies*

$$\|x^* - \bar{x}\|_2 \leq C'\|\bar{x} - \bar{x}_s\|_1/\sqrt{s} \quad \textit{and} \quad \|x^* - \bar{x}\|_1 \leq C'\|\bar{x} - \bar{x}_s\|_1$$

for some constant $C' > 0$*.*

- Why is this surprising?
  - In a traditional compression scheme, we would sample f, calculate the transform coefficients $\bar{x}$, and compress $\bar{x}$ into $\bar{x}_s$.
  - If A satisfies the assumption, using CS techniques we can compute a solution x^* of quality close to $\bar{x}_s$ from the sample y only.
- Where do we find matrices A that satisfy the RIP?
  - There are several randomized schemes for sampling matrices A that satisfy the RIP whp.
  - Given a low-coherence pair (Φ, Ψ), one can sample m columns of Φ uniformly at random. If $m \geq Cs(\log n)^4$, $A = \hat{\Phi}^\top \Psi$ satisfies the RIP whp.

INTERESTED IN MORE?

- Surveys:
  - E. Candés and M. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21-30, Mar. 2008.
  - M. Duarte and Y. Eldar, "Structured compressed sensing: From theory to applications," *IEEE Trans. Signal Process.*, vol. 59, no. 9, pp. 4053–4085, July 2011.
- Textbooks:

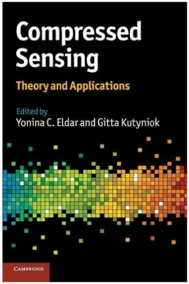


Image source: https://images-na.ssl-images-amazon.com/images/I/514fKlMG-9L._SX327_BO1,204,203,200_.jpg

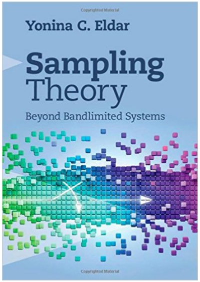


Image source: https://images-na.ssl-images-amazon.com/images/I/51Vl6MoyiwL._SX348_BO1,204,203,200_.jpg

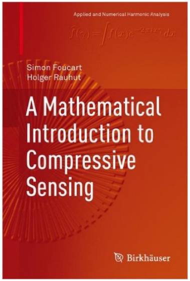


Image source: https://images-na.ssl-images-amazon.com/images/I/41s%2B50qukOL._SX331_BO1,204,203,200_.jpg

Interested in More?

- Other key references:

  - E. Candés and J. Romberg, “Sparsity and incoherence in compressive sampling,” *Inverse Prob.*, vol. 23, no. 3, pp. 969-985, 2007.
  - E. Candés, J. Romberg, and T. Tao, “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” *IEEE Trans. Inform. Theory*, vol. 52, no. 2, pp. 489-509, Feb. 2006.
  - E. Candés, J. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements,” Comm. Pure Appl. Math., vol. 59, no. 8, pp. 1207-1223, Aug. 2006.
  - E. Candés and T. Tao, “Near optimal signal recovery from random projections: Universal encoding strategies?,” *IEEE Trans. Inform. Theory*, vol. 52, no. 12, pp. 5406-5425, Dec. 2006.
  - E. Candés and T. Tao, “Decoding by linear programming,” IEEE Trans. Inform. Theory, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.
  - D. Donoho, “Compressed sensing,” *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

A Briefer Introduction to CVXOPT

What is CVXOPT?

- CVXOPT is a free convex optimization package for Python.
- It can be used with iPython or on the command line by executing Python scripts.
- It provides built-in solvers for
  - linear cone programs: `cvxopt.solvers.conelp`
  - quadratic cone programs: `cvxopt.solvers.coneqp`
  - convex programs with linear objectives: `cvxopt.solvers.cpl`
  - convex programs with nonlinear objectives: `cvxopt.solvers.cp`
- It provides routines for implementing customized solvers and interfaces to external solvers (GLPK, MOSEK, and DSDP5).
- It also provides the module `cvxopt.modeling` for modeling and solving linear programs and optimization problems with convex piecewise-linear cost and constraint functions.
- Detailed information: `http://cvxopt.org/userguide/intro.html`.

Essentials for Today

- `cvxopt.matrix` and `cvxopt.spmatrix`
  - CVXOPT extends the built-in Python objects with a `cvxopt.matrix` object for dense matrices and an `cvxopt.spmatrix` object for sparse matrices.
  - To enter a problem in matrix form into CVXOPT, data must be provided using one of these matrix objects.
  - NumPy arrays can be converted to CVXOPT matrices.
- `cvxopt.modeling`
  - Use `cvxopt.modeling.variable` to define (a vector of) variables.
  - Affine and convex piecewise-linear functions can be created with compositions of linear expressions, `max`, and `abs`.
  - Use `cvxopt.modeling.op` to create an optimization problem.
  - Call the method `cvxopt.modeling.op.solve` to solve the optimization problem: This method converts the problem to a linear program and solves it using the CVXOPT linear programming solver.

Examples

Discrete Fourier Transform (DFT)

- Let $\Psi \in \mathbb{C}^{n \times n}$ be the square unitary matrix

$$\Psi = \begin{bmatrix} \vdots & \vdots & & \vdots \\ \frac{1}{\sqrt{n}} e^{i2\pi p/n} & \frac{1}{\sqrt{n}} e^{i4\pi p/n} & \cdots & \frac{1}{\sqrt{n}} e^{i2\pi p} \\ \vdots & \vdots & & \vdots \end{bmatrix}$$

- Recall Euler's identity? $e^{ix} = \cos x + i \sin x$.
- For one-dimensional signals:
  - (Orthonormalized) DFT: $f \to \Psi^\top f$
  - (Orthonormalized) Inverse DFT: $x \to \Psi x$
- For two-dimensional signals:
  - DFT/IDFT acts on the rows first and columns later.

Discrete Cosine Transform (DCT)

- Similar to the DFT but real-valued
- Let $\Psi \in \mathbb{R}^{n\times n}$ be the square orthogonal matrix

$$\Psi = \begin{bmatrix} \vdots & \vdots & & \vdots \\ \frac{1}{\sqrt{n}} & \frac{1}{\sqrt{2n}}\cos\left(\frac{\pi(2p+1)}{2n}\right) & \cdots & \frac{1}{\sqrt{2n}}\cos\left(\frac{\pi(2p+1)(n-1)}{2n}\right) \\ \vdots & \vdots & & \vdots \end{bmatrix}$$

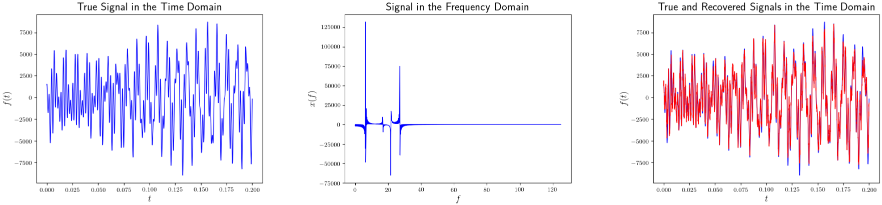
- For one-dimensional signals:
  - (Orthonormalized) DCT (of type II): $f \to \Psi^\top f$
  - (Orthonormalized) Inverse DCT (of type II): $x \to \Psi x$
- For two-dimensional signals:
  - DCT/IDCT acts on the rows first and columns later.
- DFT/DCT represents a signal as a sum of sinusoids of varying magnitudes and frequencies.
- For a "typical" sound/image signal, the sample data is correlated, and the DFT/DCT is sparse: Most of the information is concentrated in just a few coefficients of $x = \Psi^\top f$.

Example #1: Sound Sensing

- Example #1.a: Artificial sound wave

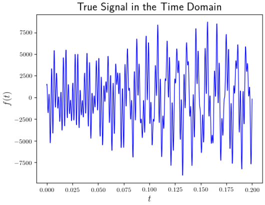
Example #1: Sound Sensing

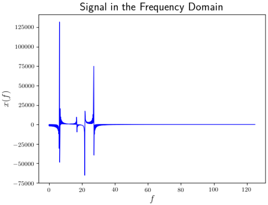
- Example #1.a: Artificial sound wave
  - Percentage sampled: 10%

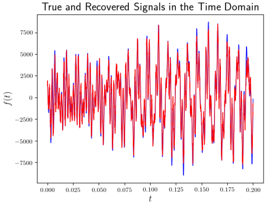


Example #1: Sound Sensing

- Example #1.a: Artificial sound wave
  - Percentage sampled: 10%



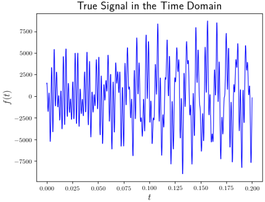


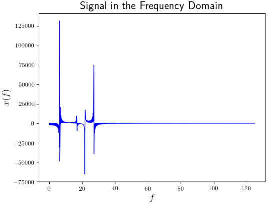


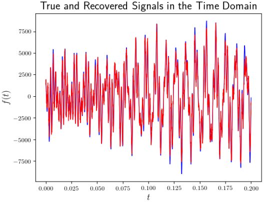
- Example #1.b: Real sound wave

Example #1: Sound Sensing

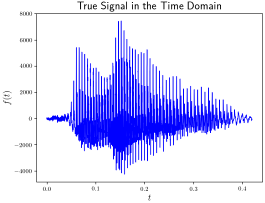
- Example #1.a: Artificial sound wave
  - Percentage sampled: 10%

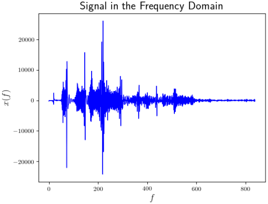


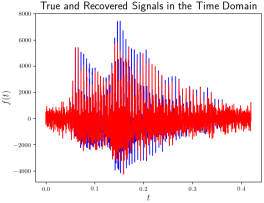




- Example #1.b: Real sound wave
  - Percentage sampled: 20%





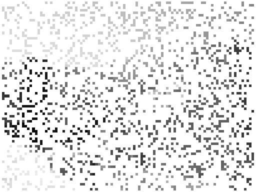


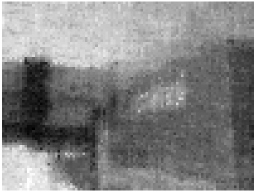
Example #2: Image Sensing

- Example #2.a: SAMSI

Example #2: Image Sensing

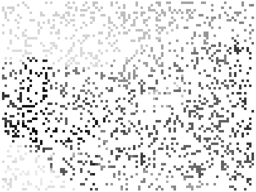
- Example #2.a: SAMSI
  - Percentage sampled: 25%

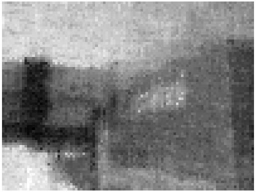




Example #2: Image Sensing

- Example #2.a: SAMSI
  - Percentage sampled: 25%

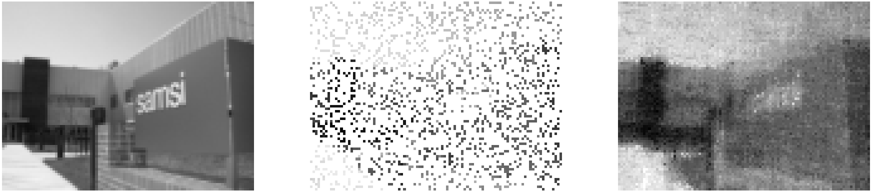




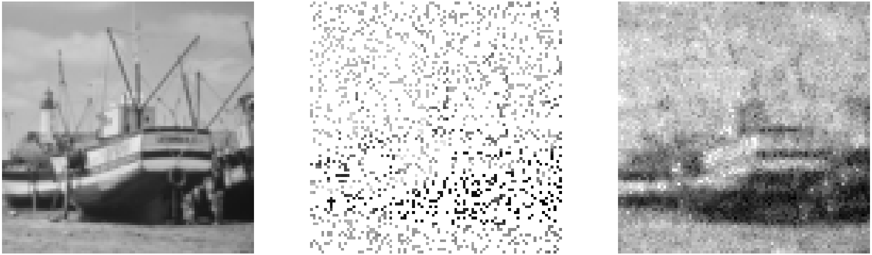
- Example #2.b: Boat

Example #2: Image Sensing

- Example #2.a: SAMSI
  - Percentage sampled: 25%



- Example #2.b: Boat
  - Percentage sampled: 25%



Questions?