Learning about Compressive Sensing using the Intel Atom E680 Processor

Bruno Vizcarra, bovhx5@umkc.edu
Walter D. Leon-Salas, PhD.
Computer Science Electrical Engineering Department
University of Missouri-Kansas City
Kansas City, Missouri
USA

Motivation

The standard sample-then-compress framework suffers from three (3) inherent inefficiencies [1]:

1) We must start with a potentially large number of samples of **N** even if the ultimate desired **K** is small.
2) The encoder must compute all of the **N** transform coefficients even though it will discard all but **K** of them.
3) The encoder faces the overhead of encoding the locations of the large coefficients.

Introduction to Compressive Sensing

The standard sample-then-compress process can be simplified via the following method [2]:

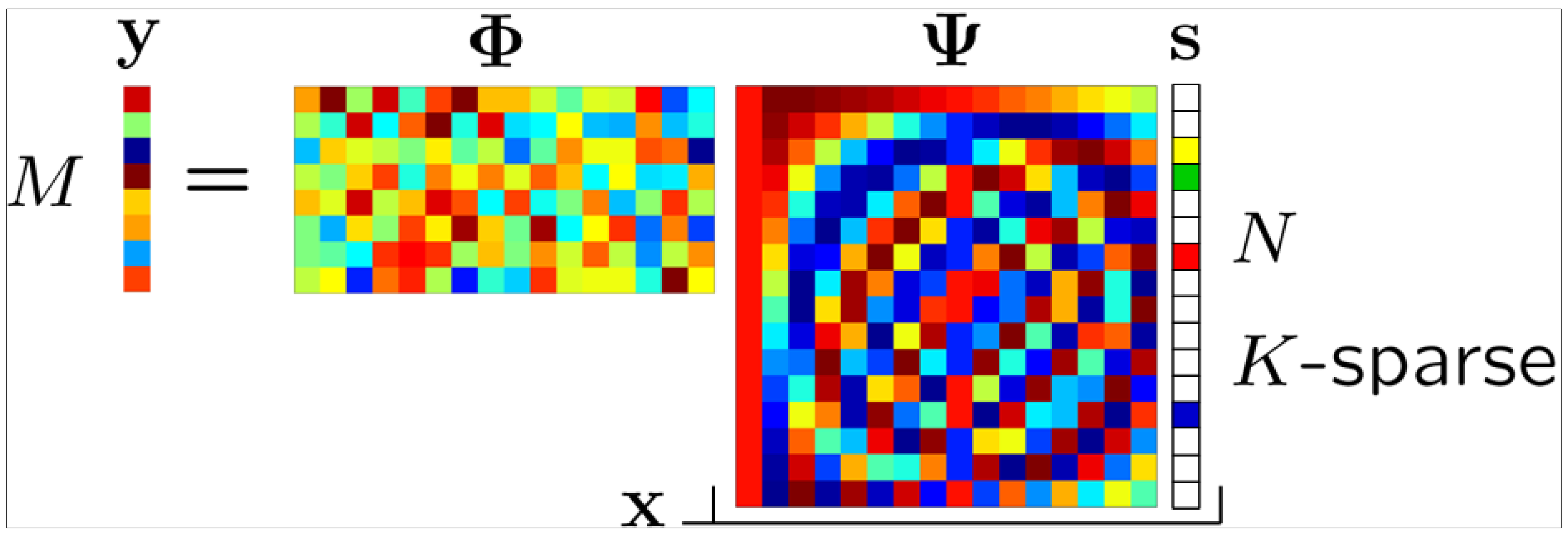
1) Directly acquire a signal in a condensed representation with **M** less than **N** measurements and place them in the vector **y**.
2) **y** is equivalent to the inner product of **x** and **M** other vectors. **x** is the **N**-sample vector from the standard sampling example.
3) If we stack these **M** other vectors into the matrix **Φ**, then the overall equation becomes:

$$\mathbf{y=\Phi x=\Phi\Psi s}$$

This operation is illustrated in Figure 1. We considered the basis matrix, **Ψ**, equal to identity.

4) Gaussian noise works well for creating the **Φ** matrix. So, the only really difficult math to be done here is in designing a reconstruction algorithm to obtain **x'** from **y**.

Figure 1



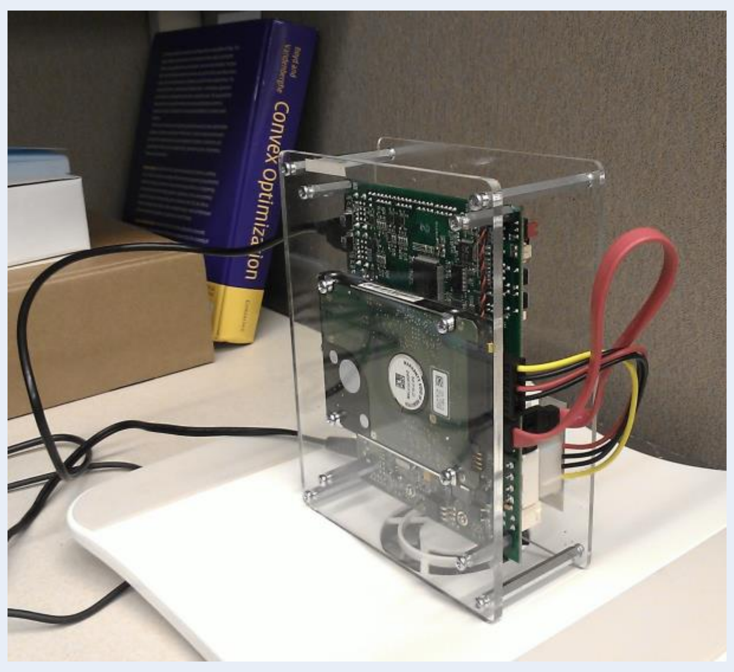
Procedure

MATLAB
y=Φx

CVXGEN

$$\min \|x\| \quad \text{subject to } y = \Phi x, \quad x \geq 0$$

Custom C Code



System setup of the Inforce IFC-9402 1GHz Portable Computing Platform

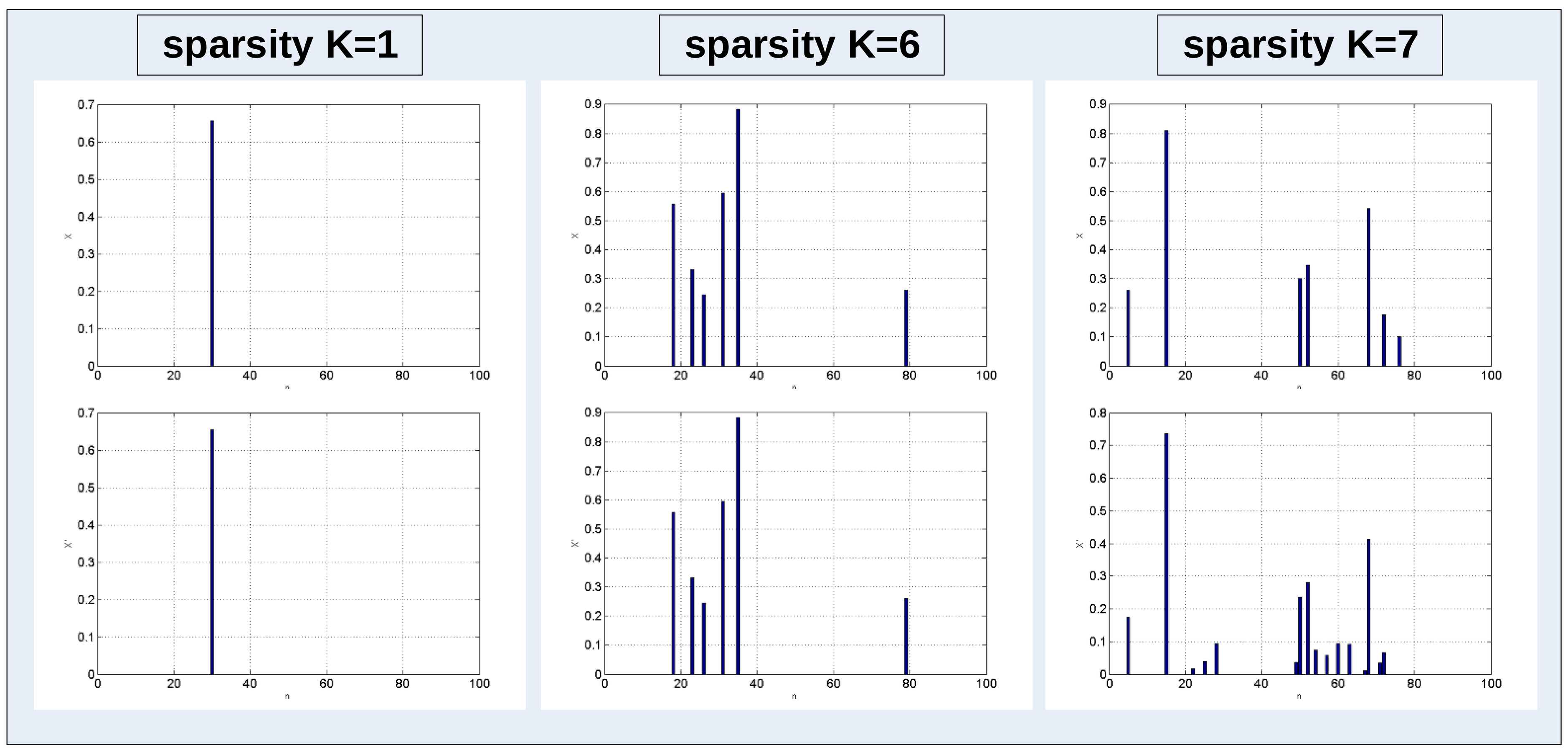
x'

1) With MATLAB we generate our **y.**
2) Using L1 optimization techniques we can exactly reconstruct **K**-sparse vectors with high probability and stability. This is a **convex optimization** problem which conveniently reduces to a linear program known as **Basis Pursuit (BP)** [3]. CVXGEN ouputs our solver code according to these linear constraints.
3) Using **y** and **Φ** as inputs, we compile and run the custom solver (complexity of $\mathbf{N^3}$) on the **IFC-9402** to generate our recovered **x'**.

Results

Table 1				K=1	K=6	K=7
				M~(CKlog(N/K))		
N	M_{actual}	Array Size	Solve time	M_1	M_6	M_7
10	3	30	0 s	2.00	2.66	2.17
100	5	500	10 milliseconds	4.00	14.66	16.17
80	16	1280	10 milliseconds	3.81	13.50	14.81
100	13	1300	10 milliseconds	4.00	14.66	16.17
70	20	1400	CCL out of memory error	3.69	12.80	14.00

Setting M equal to 2Klog(N/K) should suffice for a JPEG/JPEG2000 compressible image and in our specific case it served as a good guideline.



Conclusions and Future Work

- ❑ The original signal was recovered using compressive sensing within 10 milliseconds following the results of Table 1 above.
- ❑ The IFC-9402, complete with a Linux OS, proved as an easy-to-use tool in understanding digital signal processing techniques.
- ❑ We will implement the above procedure using Orthogonal Matching Pursuit (OMP) & Basis Pursuit with Inequality Constraints (BPIC).

[1] – Single-Pixel Imaging via Compressive Sampling by Duarte, Davenport, Takhar, Laska, Sun, Kelly and Baraniuk, IEEE Signal Processing Magazine March 2008
[2] - http://cnx.org/content/m16125/latest/

Credits: Standford Engineering for CVXGEN
Intel Corp. for Inforce 9450 Evaluation System