COMPRESSED SENSING AND ITS APPLICATIONS  
  
Done By:

DEEPTHI SUDHARSAN (CB.EN.U4AIE19022)

MEGHNA B MENON (CB.EN.U4AIE19043)



B.TECH CSE-AI, CEN DEPARTMENT, AMRITA SCHOOL OF ENGINEERING

# ACKNOWLEDGEMENT

This project would not have been possible without the support and guidance of Dr. Soman K.P, who has provided us with sufficient knowledge and guidelines even during these unprecedented and trying times.

We would also like to thank our Computer Science and Engineering (Artificial Intelligence) department for giving us this opportunity to nurture and hone our skills.

Furthermore, we would like to thank the Amrita Vishwa Vidyapeetham management for providing ample resources to avail our project needs and a platform for online lectures during this lockdown.

TABLE OF CONTENTS

[ACKNOWLEDGEMENT 2](#_Toc59890062)

[ABSTRACT 5](#_Toc59890063)

[COMPRESSED SENSING 7](#_Toc59890064)

[Why Sparsity? 7](#_Toc59890065)

[Role of L1 – Regularization 7](#_Toc59890066)

[L1 – norm Visualization 9](#_Toc59890067)

[COMPRESSED SENSING USING L1 - REGULARIZATION 10](#_Toc59890068)

[Problem 1: 10](#_Toc59890069)

[MATLAB Code: 12](#_Toc59890070)

[Problem 2: 13](#_Toc59890071)

[MATLAB Code: 13](#_Toc59890072)

[COMPRESSED SENSING USING ADMM 14](#_Toc59890073)

[MATLAB Code: 17](#_Toc59890075)

[GROUP TESTING FOR COVID-19 USING COMPRESSED SENSING 18](#_Toc59890076)

[Group Testing for Increased Testing Capacity 20](#_Toc59890077)

[How do we use Compressed Sensing to do this task? 20](#_Toc59890078)

[MATLAB Code: 20](#_Toc59890079)

[Error Correction Codes (COVID-19) 21](#_Toc59890080)

[Formulation 22](#_Toc59890081)

[Expanded Bipartite Graphs 23](#_Toc59890082)

[SIGNAL RECONSTRUCTION/RESTORATION USING COMPRESSED SENSING 24](#_Toc59890083)

[Discrete Cosine Transform: 25](#_Toc59890084)

[MATLAB Code: 26](#_Toc59890085)

[IMAGE RECONSTRUCTION USING COMPRESSED SENSING 28](#_Toc59890087)

[Python Code 29](#_Toc59890088)

[TOTAL VARIATION MINIMIZATION USING ADMM (SIGNAL DENOISING) 31](#_Toc59890089)

[MATLAB Code: 32](#_Toc59890090)

[OTHER APPLICATIONS 36](#_Toc59890091)

[REFERENCES 37](#_Toc59890092)

# ABSTRACT

Compressed Sensing is one of the most important and highly sought-after signal processing technique used for the recovery and reconstruction of signals. Compressed Sensing finds itself in many important applications like in Single Pixel Camera, MRI imaging, image recovery and preprocessing etc.

This project covers some of the important aspects of compressed sensing, how it is used for the group testing of COVID-19 during this pandemic, how we can apply the concept of ADMM (Alternating Direction Method of Multipliers) to solve the compressed sensing minimization problem and also few applications of Compressed Sensing.

Keywords:  Compressed Sensing (CS), Sparsity, ADMM, Optimization, Basis Pursuit

# COMPRESSED SENSING

**Compressed sensing** is a signal processing technique which helps in efficiently recovering, processing and reconstructing a signal (by sampling it at a rate much below the Nyquist sampling rate), by finding **sparse** solutions to underdetermined linear systems (System of linear equations, with more equations than unknowns).

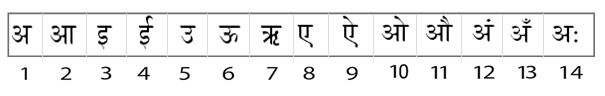
Nyquist Rate  : For effective reproduction, the sampling rate should be twice the highest frequency in the signal, this sampling rate is called as Nyquist rate. Nyquist sampling rate fs = 2W where W is the highest frequency.

## Why Sparsity?

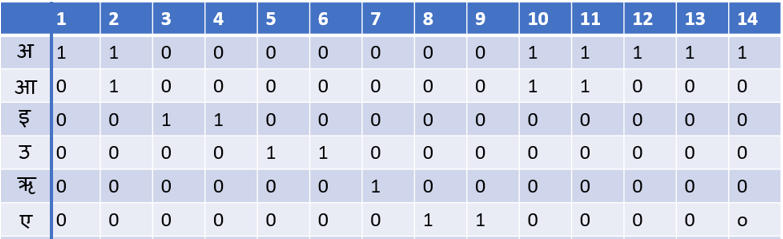
We let the constraint of sparsity come into picture so as to allow only solutions which have small number of non-zero coefficients to the underdetermined linear system. It is not necessary that all underdetermined linear system solutions need to be sparse, but if the system has a unique sparse solution, recovery of the solution is easier. If the solution is sparse, it is compressible and recovery is possible in minimum number of steps.

## Role of L1 – Regularization

Let us look at a small example for our better understanding. Suppose we have all the vowels of the Hindi language and we want to train a classifier into classifying the different vowels based on any similarity or constraint that too in a smaller number of iterations or steps. What we do is as follows. We take the training data set and numerically label all of them like below:



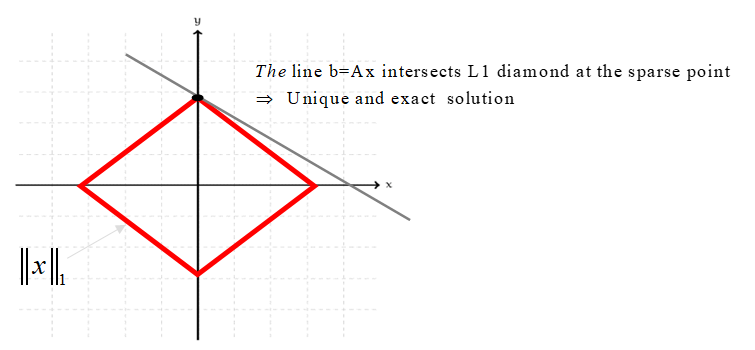
Now, our goal is to break-down the training data set into a set of smaller datasets derived from the complete training data and create/represent them as sparse vectors. For example, taking the similarity of the curves or lines into consideration, we can create this “measurement matrix”.

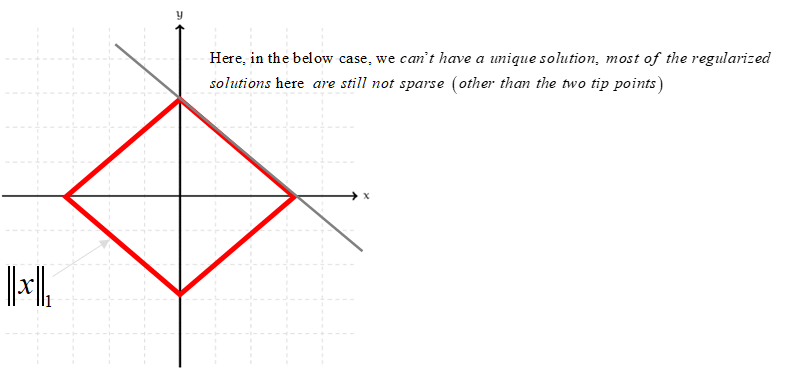


When we are solving for a large vector x with less training data ( Solving Ax=b where A matrix consists of the training data, x is the solution vector we want and b is the label vector), there can be many solutions for x, even infinite! To find good solutions amongst these solutions,  we should make sure that each component selected from solution vector x contains only important and necessary features of the data. By L1 regularization, we are basically making the x vector smaller by making them sparse (sparse vectors contain less non-zero values/useful values and many useless values (zeros) that we can ignore).

## L1 – norm Visualization

We can visualize L1-norm this way,



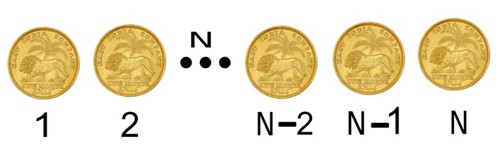


# COMPRESSED SENSING USING L1 - REGULARIZATION

## Problem 1:

Consider the given problem :

There are 7 gold coins, six of 2-grams and one coin of  1.8 grams.  How will you identify the coin with 1.8 gram with minimum number of weighing?



Note: Here, N=7.

Let us firstly number the gold coins given at hand from 1 till 7.

We will make three measurements (since we are taking 4 coins in each measurement, if we take 4-1=3 measurements to know which one is the counterfeit) with different sets of coins. We represent the combination with a sparse binary tuple vector with 0 meaning that the coin with the corresponding index number label has not been selected for the measurement and 1 meaning that the coin has been chosen in that measurement.

Let three rows of matrix A (measurement matrix) represent those vectors

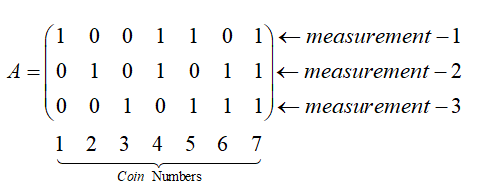
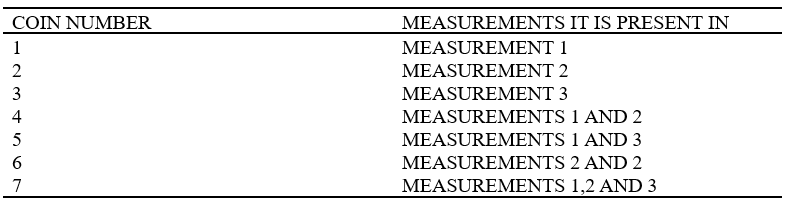


Figure : Measurement matrix

We notice a trend in the occurrence of each coin in all the measurements.



Now that we know this trend. Let us analyze the measurements that we got. Since in each measurement 4 random coins have been chosen and weighed, the total weight of the 4 coins would be 8g (2 x 2g) if they were all real gold coins. If there were any counterfeit coins, then the weight of the 4 coins would be lesser than 8g.

Analyzing the above trend and the measurements that we have gotten after weighing the coins, we can infer the following:

Coin #1 can be a counterfeit coin if the measurement 1 alone is below 8g.

Coin #2 can be a counterfeit coin if the measurement 2 alone is below 8g.

Coin #3 can be a counterfeit coin if the measurement 3 alone is below 8g.

Coin #4 can be a counterfeit coin if the measurements 1 and 2 alone are below 8g.

Coin #5 can be a counterfeit coin if the measurement 1 and 3 alone are below 8g.

Coin #6 can be a counterfeit coin if the measurement 2 and 3  alone are below 8g.

Coin #7 can be a counterfeit coin if all the measurements are below 8g.

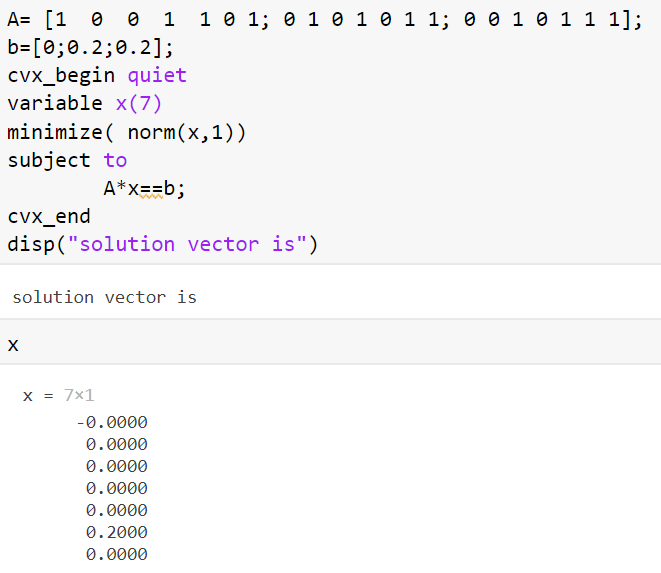
Now, we shall see how we can approach the solution using optimization theory.



Here, the vector from which we are subtracting the measurements that we have gotten is the expected measurements assuming that all the coins chosen in all the three measurements aren’t counterfeit.



### MATLAB Code:

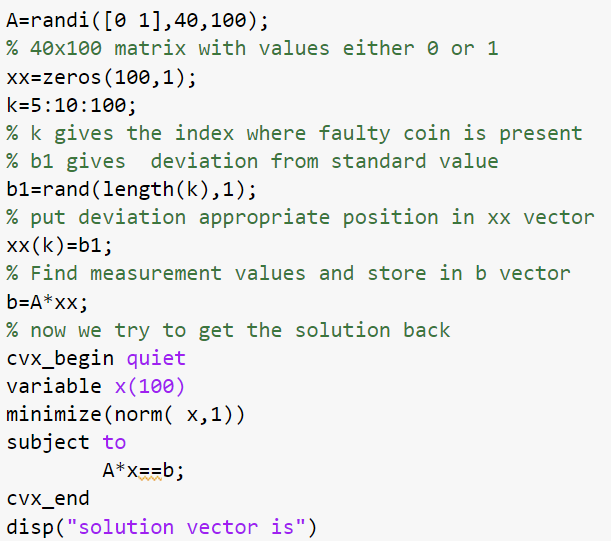


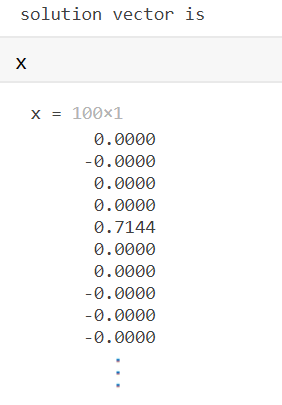
Since (8 – measurement 6 = 0.2, then the counterfeit coin is coin number 6).

## Problem 2:

Given that there are 100 coins and out of them 10 are faulty, how do we identify the coins with say 10 measurements?

### MATLAB Code:





# COMPRESSED SENSING USING ADMM

# (ALTERNATING DIRECTION METHOD OF MULTIPLIERS)

Optimization is finding the best/optimum output by maximizing or minimizing a given function such that it  also satisfies certain constraints. ADMM is a powerful iterative algorithm that solves convex optimization problems (optimization problems wherein the objective function or any of the constraints are convex in nature) by breaking them into smaller pieces. The classic ADMM splits a complex problem into subproblems which is easy to solve.

Basis pursuit is often used for finding a sparse solution to an underdetermined system of linear equations. From the role of Compressed sensing and the use of sparsity that was discussed in the previous sections, we can say that Basis Pursuit plays a central role in the theory of compressed sensing.

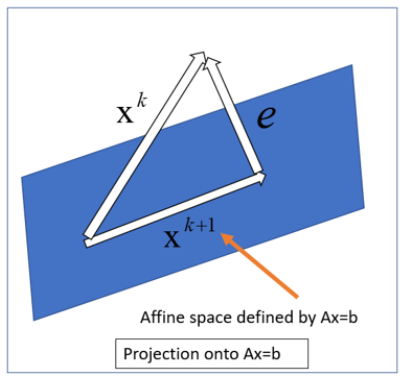
The Basis Pursuit minimization problem is given as follows:







How?





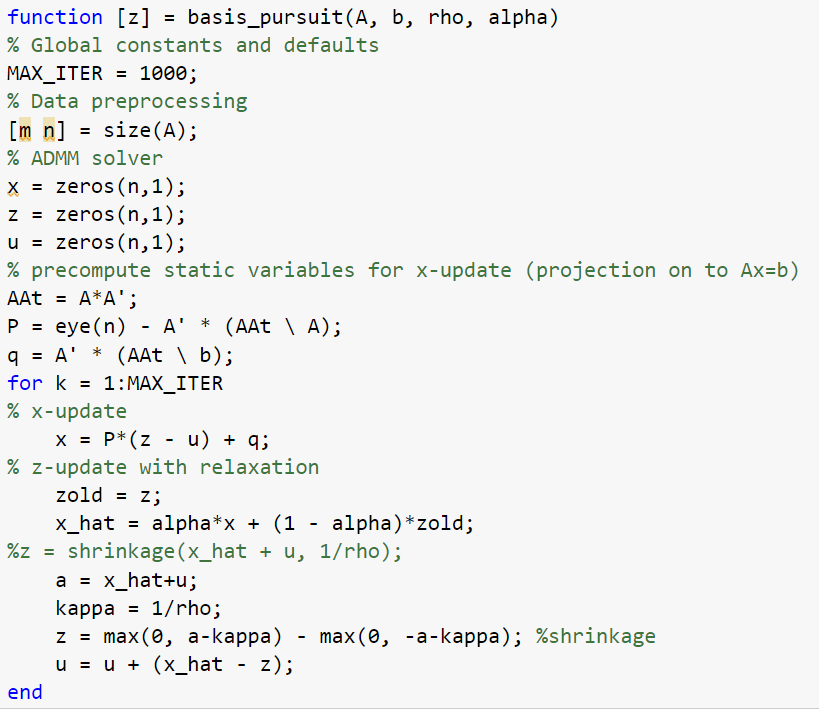


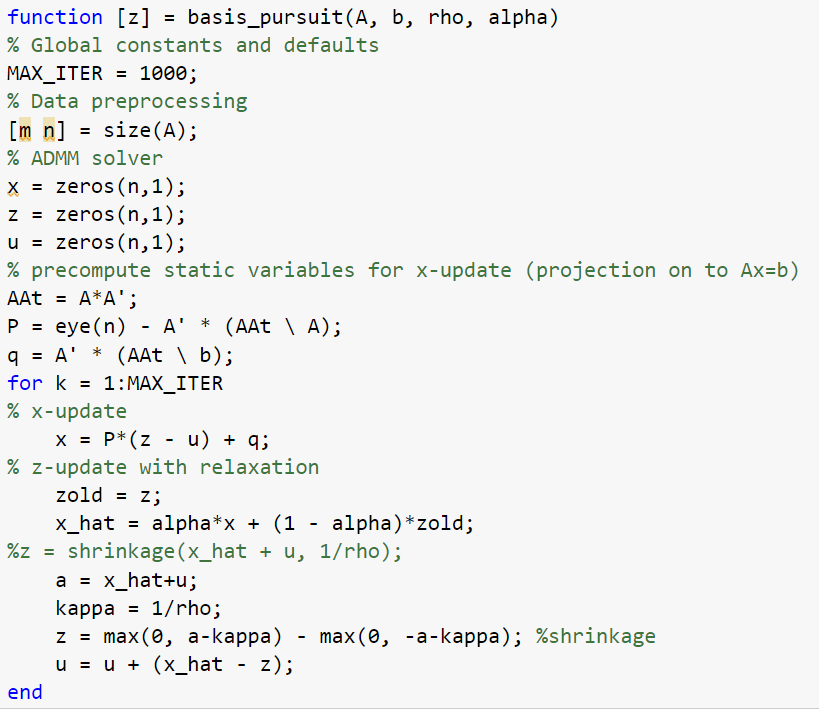


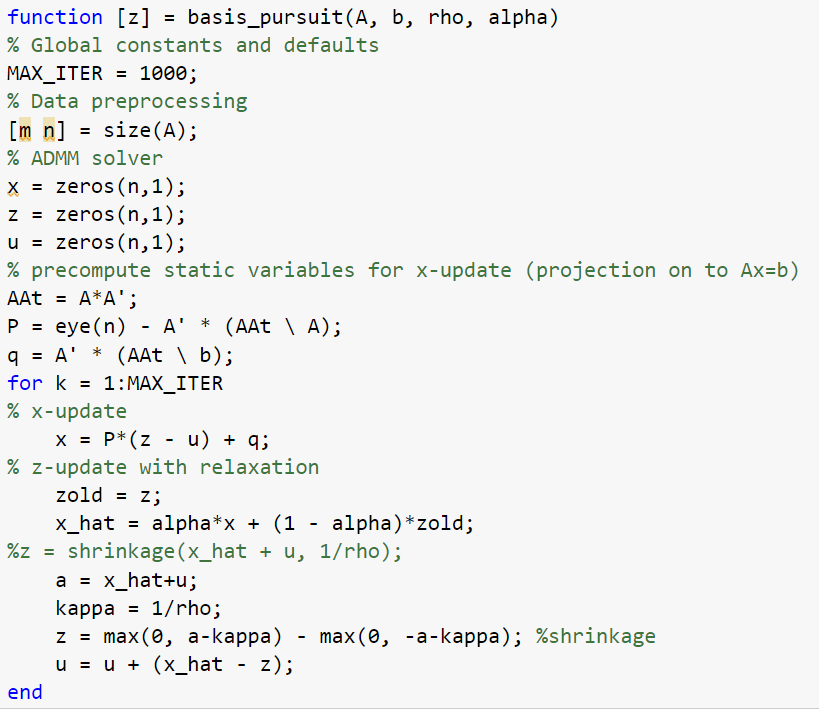


### MATLAB Code:

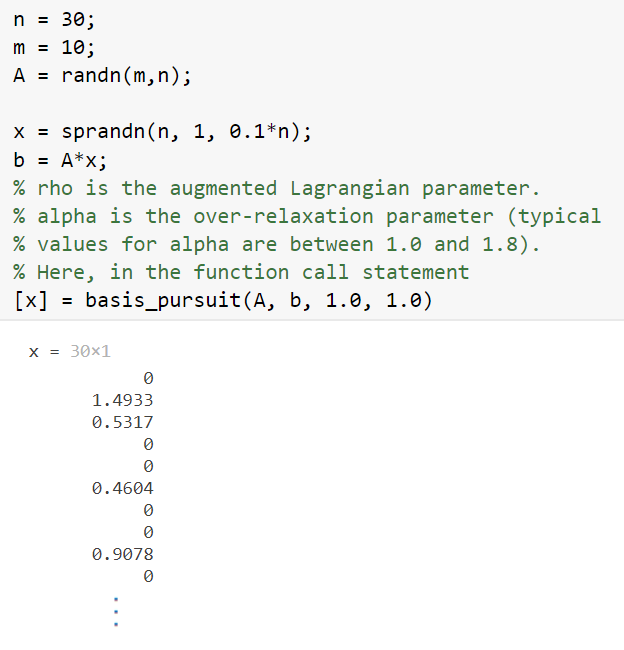
#### Function Definition







#### Creating the problem and calling the function

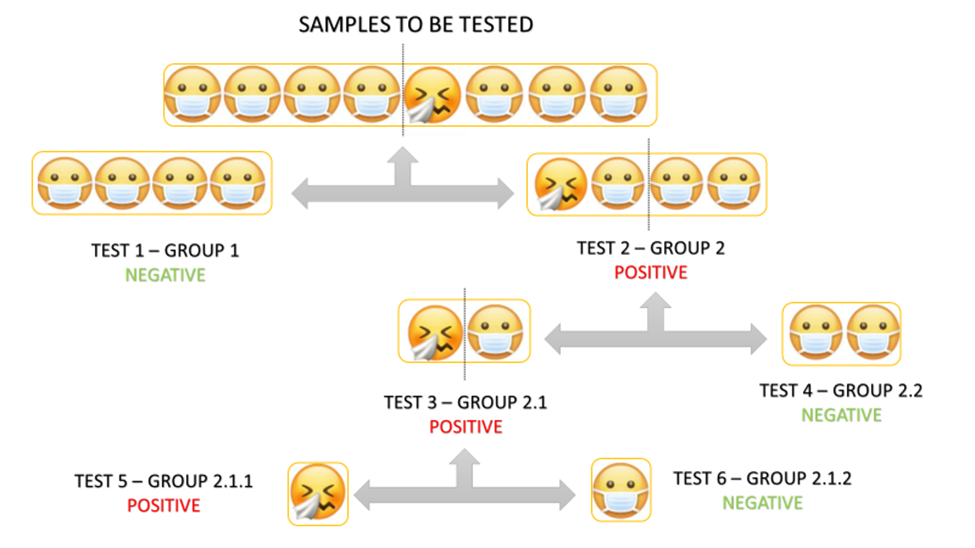


# GROUP TESTING FOR COVID-19 USING COMPRESSED SENSING

How do we apply the “Gold coin – Compressed Sensing” logic in Group Testing for COVID-19?

In order to quickly test multiple patients for the virus and to the reduce the amount of time that is spent on checking for the virus on many COVID negative patients, group testing was introduced.

Group testing works like this: First we collect the samples of the group of people who need to be tested, and we divide this sample into two groups of samples, suppose, as given in the below image, we have 8 samples that need to be checked, we divide it into two groups of 4 samples each. The group which yields a positive COVID test result is further analyzed by once again dividing the group and taking the samples of these sub groups. This goes on until we get to the sample that has tested positive.



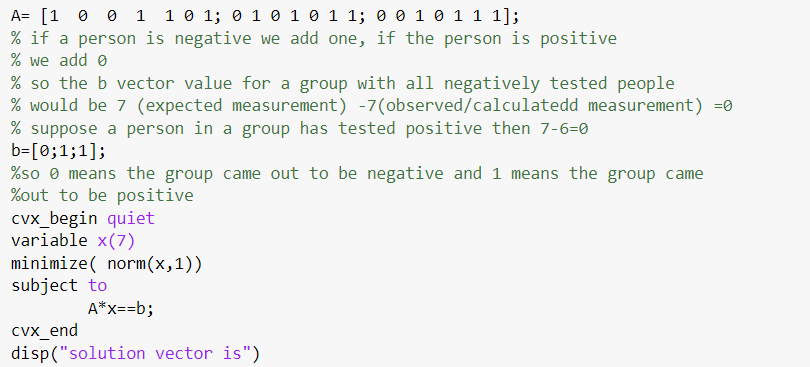
## Group Testing for Increased Testing Capacity

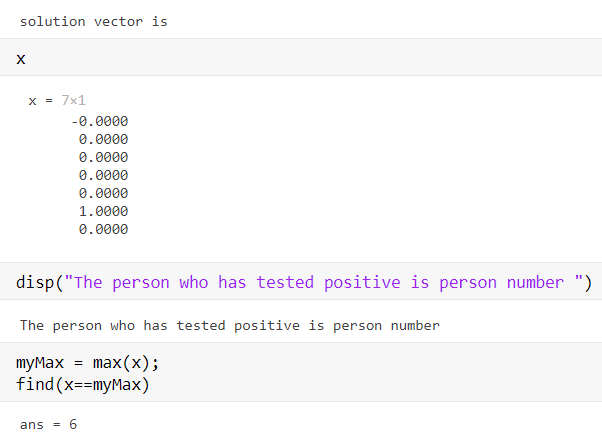
A single negative test result on a pooled sample indicates all the individuals in that pool are virus free. Pooling samples can be effective with the existing PCR tests. Compressed sending method better that group testing

## How do we use Compressed Sensing to do this task?

Just like how in the first coin problem we discussed above, instead of like the coins numbered from 1 to N, we have samples of people to be tested. Then, we take some m number of tests, and in each test we take some p number of samples. Here, our m vector can have values between 0 to 1, where 0 means that the particular test came out negative and 1 represents the test group that came positive. Then we go follow the same procedures as the coin problems and get the number/index of the sample of a positively tested person.

### MATLAB Code:





## Error Correction Codes (COVID-19)

ECC (Error Correction Codes) is very prevalent in today’s world. It is used for limiting the different errors in the data that occur over some disturbed communication channel. This limiting can be done in many different ways.

Here our main objective is to design pooled testing using 'error correction coding’ combined with compressed sensing to prove that pooled testing is not less reliable that testing individual separately. With only group testing/compressed sensing you can only reduce the required number of tests.

The main aim of error correcting pooled testing is to increase the reliability of the PCR tests and not to reduce the required number of tests as in group testing/compressed sensing. We will now do the mathematical formulation for performing the test through error correction codes.

### Formulation

So now we consider ‘n’ samples of ‘n’ people. With these ‘n’ samples, we shall perform ‘m’ PCR tests to determine the amount/quantity of the COVID-19 viruses in these given samples.

From each of these ‘m’ conducted tests, we will make a pooled sample by mixing the samples of different individuals.

Then we take a matrix which is a sparse binary matrix which will tell us the participation of the ‘n’ samples in ‘m’ tests. This means that the sample of the person will participate in the test if else it won’t i.e,

So hence the number of 1's in column of P is the number of tests that the sample of the person will participate in.

Next we have which is our weight matrix where we will be modeling the allocation of the person’s sample. The weight can be any value from 0 to 1.

Now we have to set up our measurement matrix such that:

The given ‘m’ samples will go through ‘m’ PCR tests to get the estimate the amount of DNA present in it. Due to a lot of noises and errors caused by various factors such as dilutions, sample contamination and mistakes during the testing, the final measurements from the real time PCR tests are given as:

where and are the noise from the observations and the gross error respectively.

What we need to find is (sample measurements) for n people and m tests measurements.

### Expanded Bipartite Graphs

An expanded graph is a graph that is highly sparse and has strong connectivity properties and bipartite means that the vertices of that graph can be divided into two separate disjoint and independent sets also making sure that every edge connects to vertices from both the sets only. These graphs are very commonly used for error correcting codes. So based on this graph, we design our matrix.

Also, the low commonness among the population lets us assume that is also sparse or almost sparse. It is also assumed that the gross error is sparse because things like dilution and contamination happens very rarely.

Now considering all the above assumptions we can formulate it as a constrained optimization problem as follows:

where is sum of the absolute values in . Here is used as an estimate for, as an estimate for and as an estimate for

We are using l1 norm optimization as it gives good performance without much of computation complexity.

Once we get the estimation for , if where is some threshold value, then we can say that the person is infected with the virus and tested **positive**, else we declare it as **negative**.

# SIGNAL RECONSTRUCTION/RESTORATION USING COMPRESSED SENSING

The Nyquist-Shannon sampling theorem states that to restore a signal exactly and uniquely, you need to have sampled with at least twice its frequency. Of course, this theorem is still valid; if you skip one byte in a signal or image of white noise, you can’t restore the original. But most interesting signals and images are not white noise. When represented in terms of appropriate basis functions, such as trig functions or wavelets, many signals have relatively few non-zero coefficients. In compressed (or compressive) sensing terminology, they are sparse.

When you say that it is a sparse matrix or an array, it means that most of the elements in the matrix or array are zero.

Considering a raw image/signal in terms of a vector with many components. We can assume that can be represented as a linear combination of certain basis functions :

where is the discrete cosine transform and most of the coefficients are effectively zero, so that is sparse.

## Discrete Cosine Transform:

The DCT is a technique allowing the conversion of a signal into elementary frequency components. More in particular, in the DCT the input signal is represented as a linear combination of weighted basis functions that are related to its frequency components.

In general, the DCT does not directly reduce the number of bits required to represent the block. For instance, for an 8 × 8 block of 8-bit pixels, the DCT produces an 8 × 8 of 11-bit coefficients due to the range of coefficient values. However, considering that the DCT concentrate, the low-frequency coefficients, and remaining other coefficients are mainly zero, the compression can be achieved by transmitting the near-zero coefficients and by quantizing and coding the remaining coefficients.

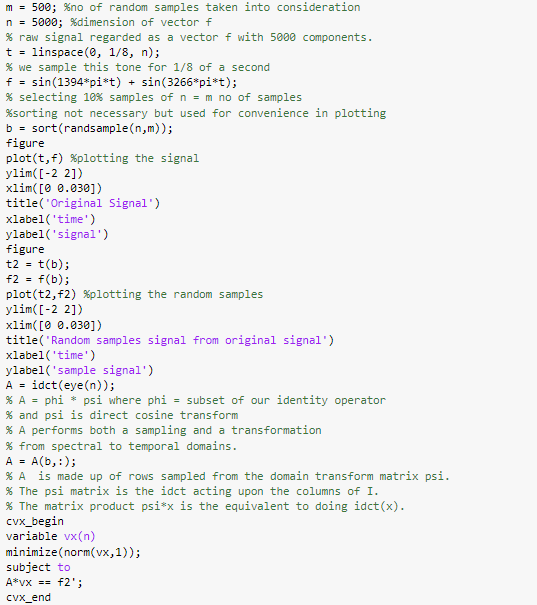
The sampling of signal involves another linear operator :

where is a few samples from and is a subset of the rows of the identity operator.

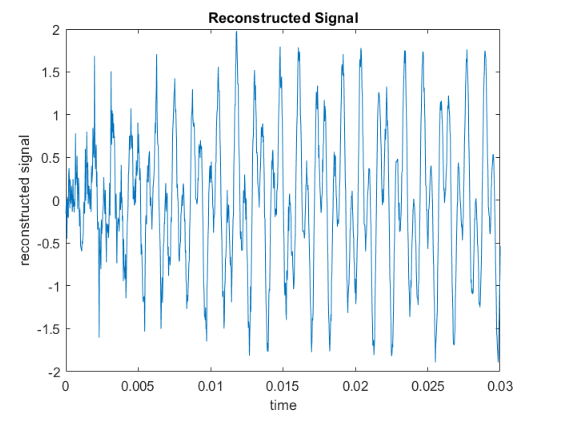
Now to reconstruct our signal, we must try to recover the coefficients by solving:

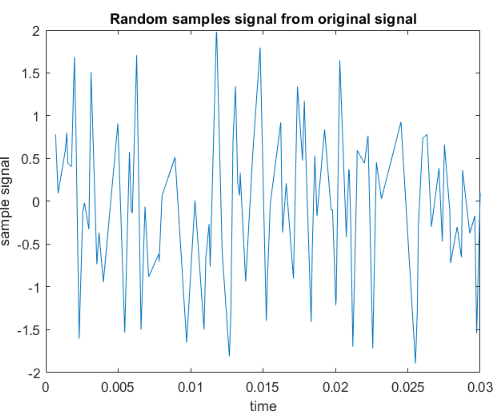
Once we have the coefficients we can recover the signal by solving/computing:

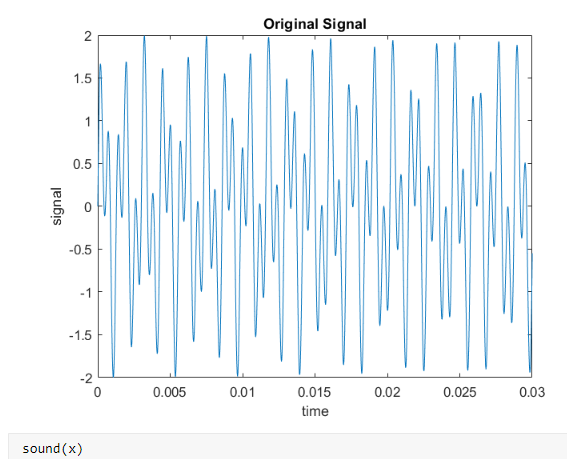
## MATLAB Code:



# 







# IMAGE RECONSTRUCTION USING COMPRESSED SENSING

What we had done previously was for a one-dimensional signal, whereas here we shall do the image reconstruction for an image in Python.

Here we have used SciPy’s and functions to act on the columns of the matrix.

Also, one major difference here in the 2D case is the generation of the matrix.

Creating the matrix for 2D image data takes a little more ingenuity than it did in the 1D case. In the derivation that follows, we’ll use the Kronecker product ⊗ and the fact that the 2D discrete cosine transform is separable to produce our operator .

The Kronecker product ⊗ is nothing but the operation on two matrices of arbitrary size resulting in a block matrix.

Then, let be an image in the spectral domain and where is the identity matrix of size . Then,

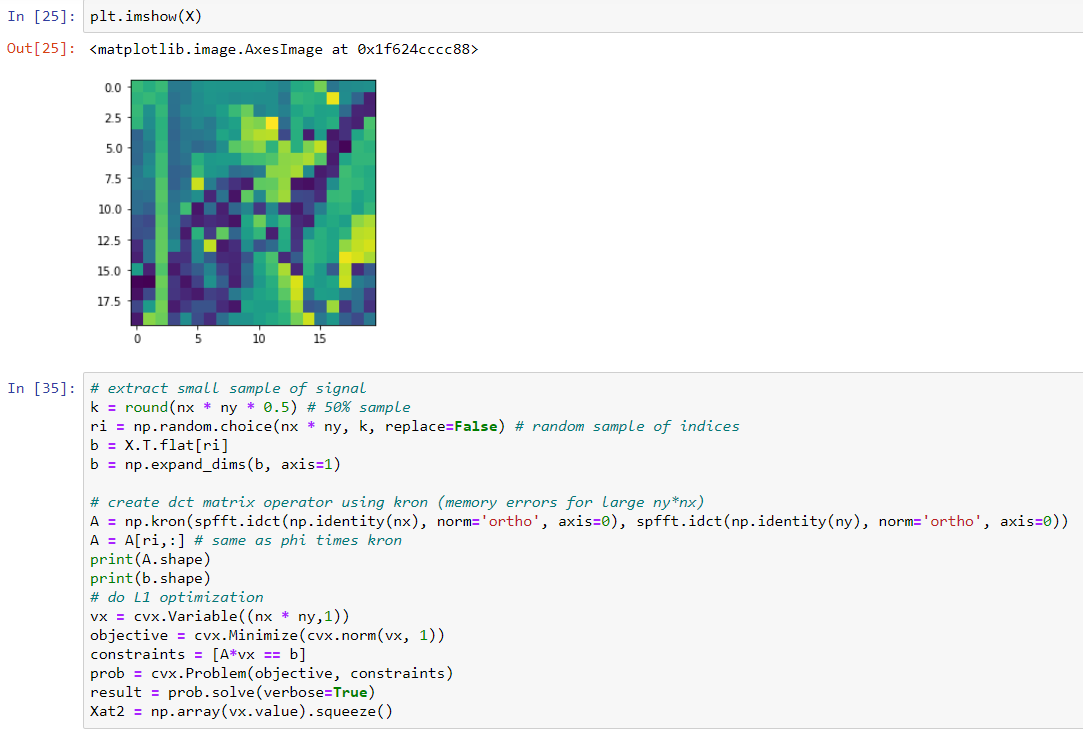
If is the vector operator that stacks columns of on top of each other, then:

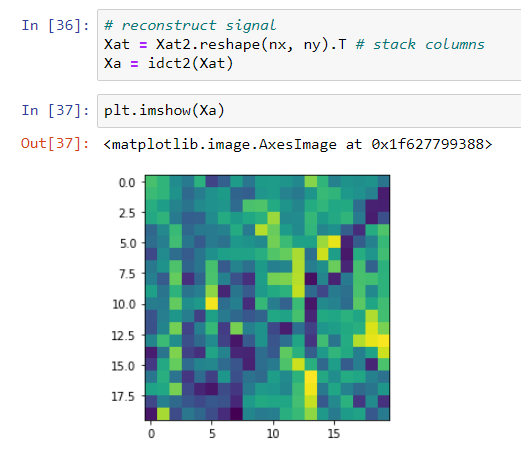
Clearly, the Kronecker product is our desired transformation matrix . Therefore, our matrix becomes , where is the sampling matrix. You can calculate the Kronecker product in Numpy with ‘numpy.kron’. The main problem with this method is that the Kronecker product can become truly massive very quickly. If your target image is and you’re taking samples, then the matrix has a size of . That said, for small images it will be fine.

## Python Code

****

****

****

****

# TOTAL VARIATION MINIMIZATION USING ADMM (SIGNAL DENOISING)

Total Variation Minimization is used in real life applications like signal denoising. Unlike our conventional Low Pass filters, the Total Variation Minimization Denoising is defined in terms of the optimization problem. Total variation is also used in applications like compressed sensing, deconvolution, interpolation etc apart from being used for denoising purposes.





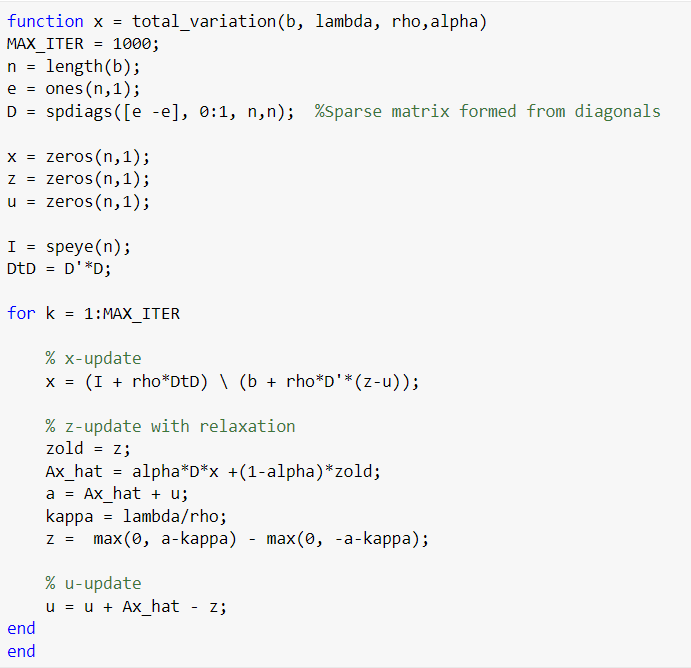


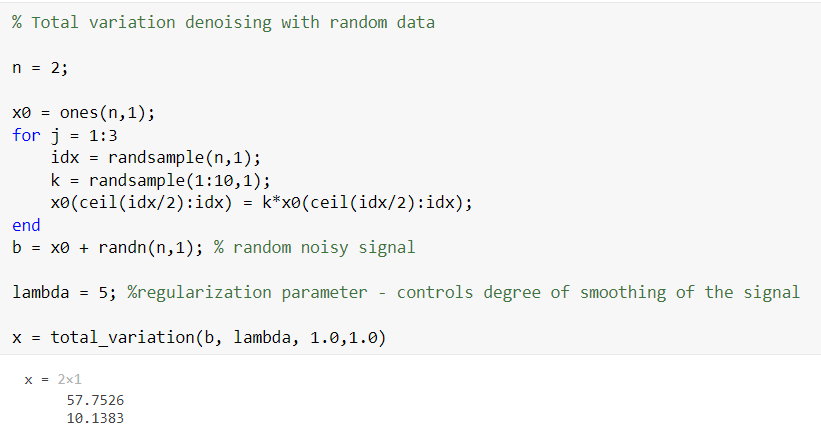






## MATLAB Code:



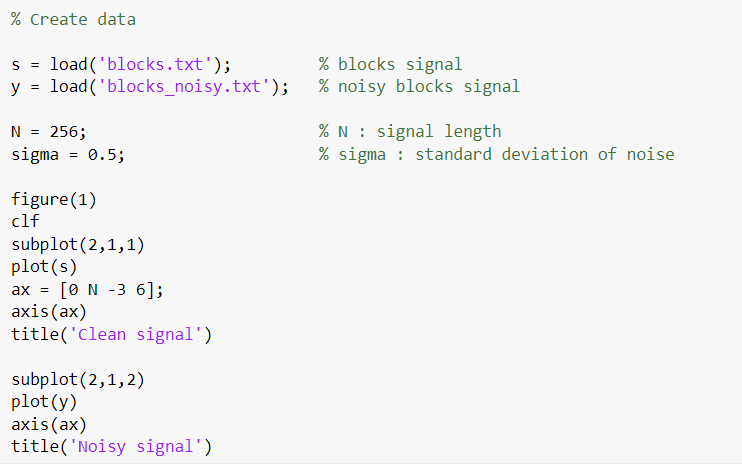


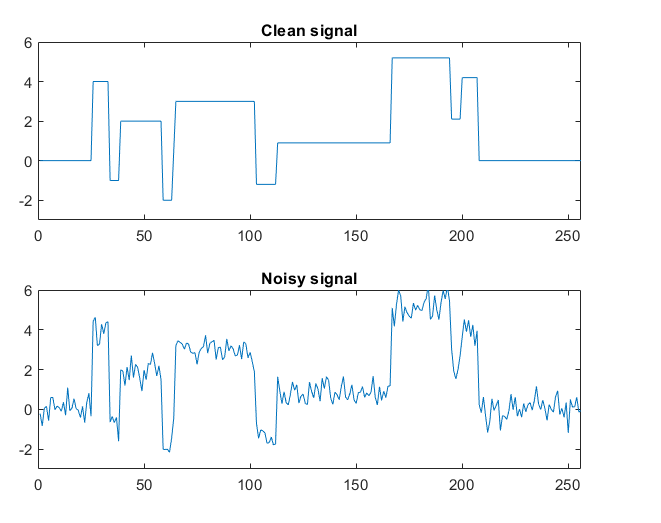
Using the similar procedure in majorization - minimization method (which is similar to ADMM; it is a powerful iterative algorithm for optimization problems), we are going to denoise a randomly generated signal.

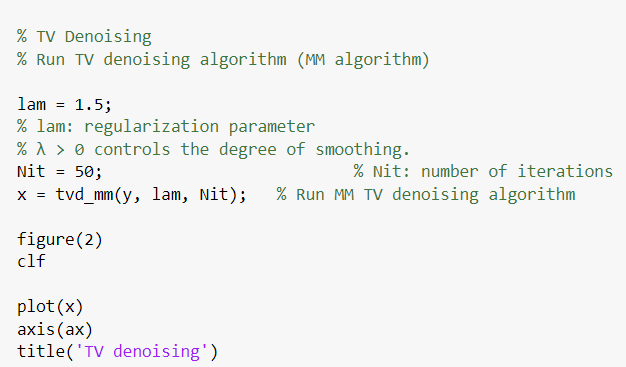
The x-update of TVD (Total variation denoising) is given to us as

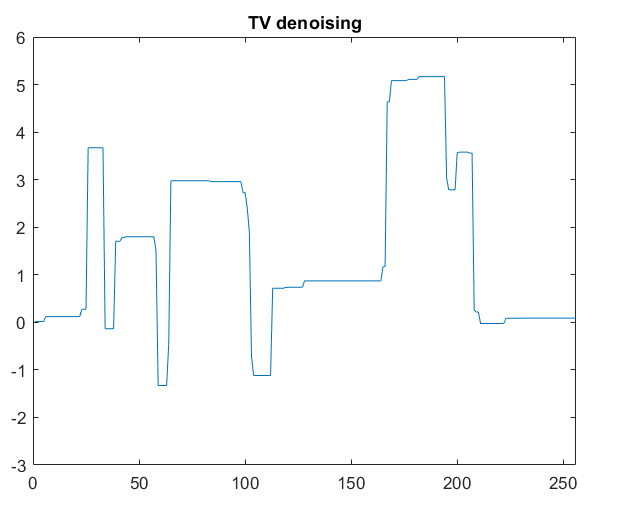


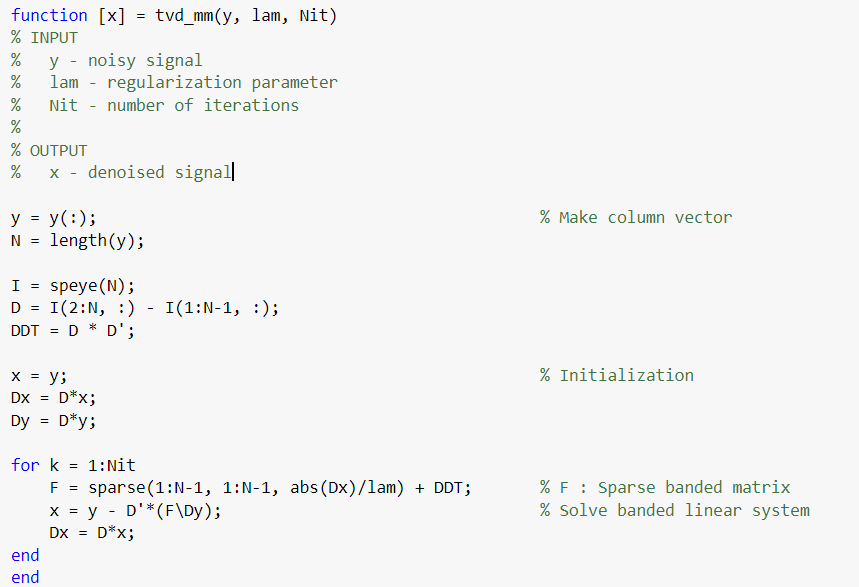












# OTHER APPLICATIONS

1. Single Pixel Camera
2. Data Compression
3. Pattern Recognition
4. Speech and Language Processing
5. Facial recognition
6. Image and signal restoration and recovery
7. Image reconstruction in holography
8. Magnetic resonance imaging

# REFERENCES

1. Group Testing for COVID-19: How to Stop Worrying and Test More - Lakshmi Narasimhan Theagarajan
2. Error Correction Codes for COVID-19 Virus and Antibody Testing: Using Pooled Testing to Increase Test Reliability - Jirong Yi, Myung Cho, Xiaodong Wu, Weiyu Xu, and Raghu Mudumbai
3. Unconstrained and Constrained Optimization Algorithms – Dr Soman K.P
4. https://medium.com/mlreview/l1-norm-regularization-and-sparsity-explained-for-dummies-5b0e4be3938a
5. https://web.stanford.edu/~boyd/papers/admm/basis\_pursuit/basis\_pursuit.html
6. Making Do with Less:An Introduction to Compressed Sensing-Kurt Bryan, Tanya Leise
7. Noisy Pooled PCR for Virus Testing - Junan Zhu, Kristina Rivera, and Dror Baron Senior Member, IEEE
8. Low-Cost and High-Throughput Testing of COVID-19 Viruses and Antibodies via Compressed Sensing: System Concepts and Computational Experiments
9. https://in.mathworks.com/company/newsletters/articles/magic-reconstruction-compressed-sensing.html
10. <http://www.pyrunner.com/weblog/2016/05/26/compressed-sensing-python/>
11. https://eeweb.engineering.nyu.edu/iselesni/lecture\_notes/TVDmm/