Abstract

This report corresponds to the evaluation of the obtained results in the fifth laboratory of Applied Harmonic Analysis.

1 Compressed sensing

In this exercise, we are going to work on the following problem:

Problem 1.1. *We have a polynomial* $p(z) = a_0 + a_1 z + \cdots + a_{N-1} z^{N-1}$ *of degree* $N-1$ *(with* N *very big) but we know that most of its coefficients are 0. That is, we know that only* K *coefficients are non zero and* K *is much smaller than* N*. Suppose that we know* N *and* K *but we don't know the polynomial. Assume that we can evaluate the polynomial at any point. How many evaluations do we need and what is a possible scheme to recover the coefficients?*

More precisely, the exercise is the following:

Exercise 1.2. *Make a program that solves problem 1 in the following sense: you hard code a sparse trigonometric polynomial (say* $p(x) = sin(x) + sin(2x) + 21sin(300x)$*) and then try to guess it by evaluating it on real points only 30 or 40 times taken at random in [0, 400]. You only know that it is a trigonometric polynomial of degree smaller than 400 and with few components.*

To address this issue, I rely on the findings of Tao, Candès, and Romberg, who have presented the following statements:

- Consider the system $Ax = y$ where A is a $N \times M$ matrix. This is the system that we want to solve (it codifies the values of the polynomial at the points that we know). This is heavily undetermined because $M < N$ so there are many possible solutions.
- Choose the solution x that has minimal ℓ^1 norm.

Therefore, let's assume the presence of a trigonometric polynomial with a degree less than 400, denoted as $p(x) = a_0 + \sum_{k=1}^{400} a_k sin(kx)$. To construct the matrix A and the vector y, I proceed by generating 35 random points within the interval [0,400]. Storing this values in $t \in \mathbb{R}^{35}$, the resulting form of the matrix A and vector y is as follows:

$$A = \begin{bmatrix} 1 & sin(t_1 x) & \cdots & sin(t_1 x) \\ \vdots & \cdots & \cdots & \vdots \\ 1 & sin(t_{35} x) & \cdots & sin(t_{35} x) \end{bmatrix}, \qquad y = \begin{bmatrix} p(t_1) \\ \vdots \\ p(t_{35}) \end{bmatrix}$$

Notice that, by solving the system $Ax = y$ and choosing the solution that has minimal ℓ^1 norm, we will find the coefficients of our polynomial. As stated in the pdf, the problem of minimizing $||x||_1$ subject to the restriction $Ax = y$ can be easily reduced to a linear-programming problem. More precisely: we define $x = u - v$ where $u_j = x_j^+$ and $v_j = x_j^-$. So we want to minimize

$$\sum_{i=1}^{N} u_i + v_i$$

subject to $A(u - v) = y$ and $u_j \geq 0, v_j \geq 0$. In Octave, the standard linear programming solver *glpk* can be employed to solve this problem. To utilize this function, we first define a vector, denoted as c. In this case, since we aim to minimize the given function, c will be a vector consisting of ones. Additionally, we include the matrix A and the vector y. By executing the function, we obtain the vector x, which possesses the smallest ℓ^1 norm and corresponds to the polynomial coefficients. Once the optimization problem concludes, the script displays on the screen the non-zero values (coefficients) of x, that coincide with those expected.