Linear programming project

Sending an encrypted message on a channel with sparse noise

Deadline: Tuesday, November 26

1 Problem description

Alice and Bob wanted to send each other encrypted messages via a channel containing sparse noise, i.e. a channel that only disturbs a small number of the message inputs, but the disturbed inputs are very strong. To be more precise, Alice wants to send a binary message $x \in \{0,1\}^p$ to

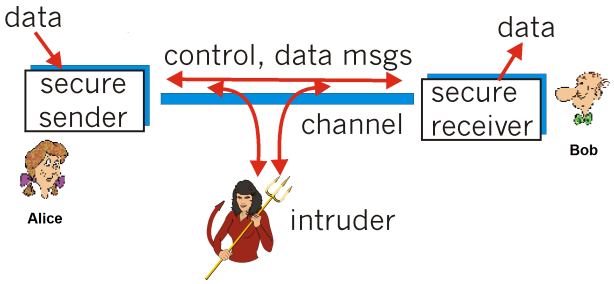

Figure 1: Problem illustration.

Bob. Before starting their communication, Alice and Bob have met and agreed on the choice of an encoding matrix $A \in \mathbb{R}^{m \times p}$ where $m \geq p$ (we'll use $m = 4p$ for this project). Alice encodes the message using the matrix A and sends the message $y = Ax \in \mathbb{R}^m$ on the channel. The channel will transmit the noisy message $y' = Ax + n$ to Bob where the noise vector n contains only a small number of non-zero inputs (e.g. 10%). When dealing with this type of noise, a good approach is to[1] minimize the ℓ_1-norm error. In mathematical terms, in order to recover Alice's message, Bob will have to solve the following optimization problem:

$$\min_{x' \in \mathbb{R}^p} ||Ax' - y'||_1 \quad \text{such that} \quad x' \in \{0,1\}^p,$$

where $||z||_1 = \sum_{i=1}^m |z_i|$ for $z \in \mathbb{R}^m$. Unfortunately, this problem is combinatorial and difficult to solve. In practice, it is common to use the following continuous relaxation

$$\min_{x' \in \mathbb{R}^p} ||Ax' - y'||_1 \quad \text{such that} \quad 0 \leq x' \leq 1, \tag{1}$$

and round off the resulting solution. If the noise is not too big, then $x' \approx x$, allowing Bob to retrieve Alice's message.

The aim of this project is to study problem (1), and thus to be able to decode Alice's messages.

[1] For those interested, see the paper *Candès and Tao, Decoding by Linear Programming, IEEE Transactions on Information Theory, 2005.*

2 Questions

1. Model the problem as a linear program. Explain your reasoning.
2. Write this linear problem in *standard* form.
3. Use the function *linprog*[2] from the Python scientific computation library ***SciPy*** to decrypt the message provided on the course website (messageFromAlice.mat). What is the message sent by Alice?
4. Is the solution obtained a vertex of the corresponding polyhedron? Justify your answer.
5. Now generate a message yourself: up to what level of noise can your message be decrypted. (i.e. how many entries of y' can be disturbed)? Is this surprising? Please comment briefly.
6. Now implement the Dikin's method presented at the end of the first part of the course to solve problem (1). Compare the results obtained between Dikin's method and the linear solvers provided by *SciPy*.
7. Use *SciPy*'s linear solver by imposing binary variables: can you decipher your message with a higher noise level?

Instructions.

1. The project is carried out in groups of 2.
2. The project is created in a Colab file or a Jupyter Notebook. Some of the code needed to carry out your project is provided here :).
3. Please enclose the Python codes within the report, and provide an .ipynb file with your names, group number in the first cell.
4. The report should not exceed 10 pages (excluding appendices containing your codes).
5. All of this has to be uploaded on Moodle by November 26.

[2]For those interested, the most efficient method proposed in "linprog" *SciPy* function is called "highs", here are some references about it: here and here