AIMS Mathematics, 10(2): 4139–4152.
DOI: 10.3934/math.2025192
Received: 16 November 2024
Revised: 17 February 2025
Accepted: 20 February 2025
Published: 27 February 2025

https://www.aimspress.com/journal/Math

Research article

Decoding as a linear ill-posed problem: The entropy minimization approach

Valérie Gauthier-Umaña[1,*], Henryk Gzyl[2,*] and Enrique ter Horst[3]

[1] Systems and Computing Engineering Department, Universidad de los Andes, Bogotá, Colombia
[2] Center for Finance, IESA School of Business, Caracas, Venezuela
[3] School of Management, Universidad de los Andes, Bogotá, Colombia

* **Correspondence:** Email: ve.gauthier@uniandes.edu.co, henryk.gzyl@iesa.edu.ve.

Abstract: The problem of decoding can be thought of as consisting of solving an ill-posed, linear inverse problem with noisy data and box constraints upon the unknowns. Specifically, we aimed to solve $\boldsymbol{Ax} + \boldsymbol{e} = \boldsymbol{y}$, where $\boldsymbol{A}$ is a matrix with positive entries and $\boldsymbol{y}$ is a vector with positive entries. It is required that $\boldsymbol{x} \in \mathcal{K}$, which is specified below, and we considered two points of view about the noise term, both of which were implied as unknowns to be determined. On the one hand, the error can be thought of as a confounding error, intentionally added to the coded message. On the other hand, we may think of the error as a true additive transmission-measurement error. We solved the problem by minimizing an entropy of the Fermi-Dirac type defined on the set of all constraints of the problem. Our approach provided a consistent way to recover the message and the noise from the measurements. In an example with a generator code matrix of the Reed-Solomon type, we examined the two points of view about the noise. As our approach enabled us to recursively decrease the ℓ_1 norm of the noise as part of the solution procedure, we saw that, if the required norm of the noise was too small, the message was not well recovered. Our work falls within the general class of near-optimal signal recovery line of work. We also studied the case with Gaussian random matrices.

Keywords: ill-posed inverse problems; decoding as inverse problem; convex optimization; gaussian random variables
Mathematics Subject Classification: 15A29, 65F22, 65J50, 46N99

1. Introduction

As we use the coding-decoding setup as motivation, we devote a few lines to that problem. Coding theory consists of the analysis of methods to encode, transmit, and recover the transmitted information. Encoding means restructuring the message, possibly increasing its size, to be able to compensate for possible transmission errors in the decoding process. In this paper, we suppose that the coding is a

linear mapping on the set of possible messages. During the transmission-recovery stage, it is here where errors come in. In this work, we refer to the input or transmitted message, and to the received output, that is to the data, as the signal. Our aim is to provide a procedure to recover the message from a, possibly noise corrupted, signal.

In cryptography, we want to send an information vector in such a way that only the intended receiver can understand it. A nice account of the multiple ways to encrypt and decrypt, since ancient times, is presented in Bauer's [1]. In public key cryptography, the idea is to use a trapdoor one-way function based on a hard problem. Until now, the cryptosystems used factoring problems and discrete log problems. Nevertheless, since the 90s, Shor's algorithm has enabled us to solve these two problems using a quantum computer. In post-quantum cryptography, the goal is to find the one-way function, based on an NP-complete problem, so that it cannot be solved using a quantum computer. McEliece [16] proposed a cryptosystem based on error-correcting codes. The encryption process includes encoding the message and adding a confounding error.

We are not concerned with the important number-theoretical and combinatorial problems arising when encrypting and decrypting a message. Instead, we deal with the decoding problem in the spirit of Candes and Tao in [4] and [5], who regard it as a problem of recovering a signal from a noisy message corrupted by noise using ordinary, real number arithmetics.

To correct or compensate for the transmission error, a linear operation is applied to the message (signal) vector $\boldsymbol{f} \in \mathbb{R}_+^m$ using an $m \times n$ generator matrix $\boldsymbol{G}$ with positive entries, in which $n > m$. We assume that the noise vector $\boldsymbol{e} \in \mathbb{R}_+^n$ has a limited Hamming weight t (in our case, this will become the ℓ_1 norm). The relationship between the message and the received word is written as:

$$\boldsymbol{y}^t = \boldsymbol{f}^t\boldsymbol{G} + \boldsymbol{e}^t. \tag{1.1}$$

A word about terminology and notations. We think of vectors as columns, but state (1.1) in row form to follow the same notations as in many works on coding theory, where the superscript ^{t} denotes transposition. We later switch to the standard conventions.

In this work, we combine two points of view about the "noise" vector $\boldsymbol{e}$. On the one hand, it can be thought of as confounding noise added by the message sender and provides a decoding algorithm good enough to determine the noise and read the message correctly. In this case, for the approach that we propose below to work, we suppose that the sender tells the decoder the ℓ_1 norm of the noise. In this case, we are in the scenario of code-based cryptography.

On the other hand, the noise can be thought of as measurement noise that is caused by the transmitter/receiver system (transmission channel). In this case, the vector $\boldsymbol{e} \in \mathbb{R}_+^n$ is unknown to the sender and the receiver of the message. Of course, since both situations can occur simultaneously, it would be nice to have an algorithm that recovers the signal (message) and the confounding noise and estimates the system noise. Our proposal can be extended to cover that case.

The approaches that have been proposed to solve this problem involve a variety of mathematical tools, including number theory, combinatorics, graph tehory [7], linear and integer programming [4, 8, 17], and probability. As a small sample of a large body of literature, consider, [3, 6, 13], and the thesis [11], plus the large list of references cited in them. Consider, for example, [8] in which the authors propose a way to design a polytope containing all valid codewords, and an objective function for which the maximum likelihood codeword is the optimum point with integral coordinates. On the use of the notion of entropy in cryptography, see [20] and [15]. In [10]

the reseachers present a method to convert the maximum-likelihood soft-decision decoding problem for linear block codes into a graph search problem where the generalized Dijkstra's algorithm can be applied to the decoding procedure. Kaneko et al. propose a new soft-decision maximum-likelihood decoding algorithm, which generates a set of candidate codewords using hard-decision bounded-distance decoding [14]. In [12], you can find a review and the way the authors categorize decoding methods based on mathematical programming approaches for binary linear codes over binary-input memoryless symmetric channels. Recent advancements in physics-informed neural networks, such as the gradient-enhanced approach by [18], provide alternative methods for solving wave equation-related inverse problems, complementing our entropy minimization approach. Furthermore, our approach to handling noise in decoding problems parallels methods used in the optimal control of stochastic differential equations with random impulses, as discussed by [19].

Since our approach follows that of [4], we briefly recall their setup using the notation of (1.1). On the one hand, the intuitive approach consists of transforming (1.1) into the optimization problem

$$\text{Find} \quad min_{\boldsymbol{x}\in\mathbb{R}^n}\{\|\boldsymbol{y}-\boldsymbol{x}^t\boldsymbol{G}\|_{\ell_1}\}. \tag{1.2}$$

This needs to be determined, and $\boldsymbol{f}^*$ that minimizes the misfit between all possible transmitted signals (the range of $\boldsymbol{G}$) and the received message in some appropriate distance. Here, $\|\boldsymbol{v}\|_{\ell_1} = \sum_i |v_i|$.

Let $\boldsymbol{F}^t$ be a matrix such that $\boldsymbol{G}\boldsymbol{F}^t = 0$. Multiplying both sides of (1.1)$\boldsymbol{F}^t$ to obtain $\tilde{\boldsymbol{y}}^t = \boldsymbol{e}^t\boldsymbol{F}^t$. In [4], it is shown that (1.2) is equivalent to

$$\text{Find} \quad min_{\boldsymbol{d}\in\mathbb{R}^n}\{\|\boldsymbol{d}\|_{\ell_1} : \boldsymbol{d}^t\boldsymbol{F}^t = \tilde{\boldsymbol{y}}^t\}. \tag{1.3}$$

The equivalence means that when any of these problems has a unique minimizer so does the other. Then, they prove that the noise vectors obtained as solutions of (1.3) exist, and have a minimal number of nonzero components for a large class of matrices.

To transform the notations to standard notations in inverse problems, we introduce the following symbols:

$$\boldsymbol{A} = [\boldsymbol{G}^T\ \mathbb{I}_n]; \quad \text{and} \quad \boldsymbol{\xi} = \begin{pmatrix}\boldsymbol{f}\\ \boldsymbol{e}\end{pmatrix}. \tag{1.4}$$

Thus, $\boldsymbol{A}$ is an $n \times (m+n)$ matrix and $\boldsymbol{\xi}$ is an $n+m$ vector. We use $\mathbb{I}_n$ to denote the $n \times n$ identity matrix. An important step in the procedure is the following. Since the symbols in the code as well as the errors, are taken from a known collection of numbers, and since both sides of (1.1) can be divided by a number M without altering the solution, (say, M equals the largest possible number in the collection multiplied by n), we may suppose the $\boldsymbol{x} \in \mathcal{K} = \prod_{j=1}^{(m+n)}[a_j, b_j]$ with $[a_j, b_j] \subset [0,1]$. This choice provides us with added flexibility that enables us to incorporate prior knowledge about the unknown code or the noise added to the confound. If no such knowledge is available, one may put $a_j = 0$ and $b_j = 1$ for $j = 1, ..., m+n$.

We can state the decoding problem as:

$$\text{Solve} \quad \boldsymbol{A}\boldsymbol{\xi} = \boldsymbol{y}; \tag{1.5}$$

$$\text{Subject to} \quad \boldsymbol{\xi} \in \mathcal{K}. \tag{1.6}$$

Observe that this problem consists of determining a vector in $\mathbb{R}^{m+n}$, satisfying box constraints from using data as a vector in $\mathbb{R}^n$, that is, of solving a convex constrained system of n equations with $m+n$ unknowns: The first m of which is the unknown message and the last n is the noise.

We mentioned two possible interpretations of the error term in (1.5). In the new notations for the inverse problem to be solved, within the spirit of (1.2), the additive noise in (1.5) can be interpreted as a slack variable that absorbs the misfit between the true signal $\boldsymbol{G}^t\boldsymbol{f}$ and the observed signal $\boldsymbol{y}$. Thus, the proposal in [4] amounts to a criterion to choose among the infinitely many solutions.

However, if the error term is used by the message transmission system to confound the data, from the point of view of the inverse problem, the noise term plays the role of a superfluous part of the message to be disregarded after the message is decrypted. Below, we see that when the ℓ_1-norm of the confounding noise is provided as part of the signal, the full message can be properly decoded. This is incorporated into the restatement of the problem as follows:

$$\boldsymbol{A}_e = \begin{bmatrix} \boldsymbol{G}^T & \mathbb{I}_n \\ \boldsymbol{0} & \boldsymbol{u} \end{bmatrix}, \quad \boldsymbol{r}_e = \begin{pmatrix} \boldsymbol{y}^T \\ s \end{pmatrix}. \tag{1.7}$$

Here, $\boldsymbol{0}$ is a row vector of zeros of size m and $\boldsymbol{u}$ is a row vector of ones of size n. The resulting matrix $\boldsymbol{A}_e$ has size $(n+1)\times(n+m)$. Also, the new data vector $\boldsymbol{r}_e$ is a column vector of size $m+1$, where the last component is $s = \boldsymbol{e}\boldsymbol{u}^T = \sum_{i=1}^{m} e_i$. With these notations, instead of problems (1.5) and (1.6), we have

$$\text{Solve} \quad \boldsymbol{A}_e\boldsymbol{\xi} = \boldsymbol{r}_e; \tag{1.8}$$

$$\text{Subject to} \quad \boldsymbol{\xi} \in \mathcal{K}. \tag{1.9}$$

The difference between (1.5)–(1.6) and (1.8)–(1.9) is the extra datum s. This datum plays an important practical role: It prevents the optimization algorithm from assigning values to the error term to minimize the misfit between $G^T\boldsymbol{f}$ and $\boldsymbol{y}$. The other role of s is that of control parameter to minimize the ℓ_1-norm of the error. Since the components of $\boldsymbol{e}$ are positive, the constraint $\langle \boldsymbol{u}, \boldsymbol{e}\rangle = s$ is equivalent to fixing the ℓ_1-norm of $\boldsymbol{e}$, and decreasing s amounts to decrease the norm of the error.

It is our objective to propose a method to solve either (1.5)–(1.6) or its extended version (1.8)–(1.9). Instead of minimizing the misfit $\boldsymbol{y} - G^t\boldsymbol{f}$, we transform (1.8)–(1.9) into a variational problem (2.1), where the objective function is an entropy of the Fermi-Dirac type, subject to (1.5)–(1.7) as linear constraints, and such that the unknowns have to satisfy the box constraints (1.6).

In Section 2, we study the basic properties of the objective function and obtain an explicit representation of the solution to (2.1) in terms of Lagrange multipliers. This solution provides us with the decoded message as well as the noise in the measurement. The objective function is tailored to guarantee that the solution automatically satisfies the box constraints, and the vector of Lagrange multipliers is the point at which the Fenchel-Lagrange dual of the objective function achieves its maximum. One important by-product of this approach, is that the norm of the gradient of the Fenchel-Lagrange dual of the objective is a measure of the error in the reconstruction. This norm will play a key role in the second example.

In Section 3, we consider some numerical examples. In the notation of (1.1), our goal is to recover $\boldsymbol{f}$ and $\boldsymbol{e}$ from the knowledge of $\boldsymbol{y} \in \mathbb{R}^n_+$ and $\boldsymbol{G}$.

We examine the effect of varying the size of s, and we see that if the data $\boldsymbol{y}$ is originally contaminated with a confounding error, then decreasing the value of s, leads to poor reconstructions.

The data for the first example is produced as if for a real coded message. The components of the message and those of the noise vector, are natural numbers in the range $[0, 10]$. They are rescaled by

a factor of $1/100$ to bring them down to $[0, 1]$, and after the solution is obtained, they are scaled back to their initial range. For the first example, we consider the ℓ_1-norm of the noise vector to be known to the decoder.

For the second example, we use the same data as the first, except that we will use the norm of the error term as a control parameter, and decrease it for each iteration of the entropy minimization algorithm. This is equivalent to an entropy minimization approach to an interior point approach to a linear programming problem. The mathematics of the problem was considered in [9] in a somewhat different setup. We see that if the data $\boldsymbol{y}$ is originally contaminated with a confounding error, then decreasing the value of s, leads to poor reconstructions. To double-check the procedure, we apply the technique into a collection of Gaussian random matrix codes.

2. The entropy minimization approach

Problems (1.5) and (1.6) are an ill-posed linear inverse problem with convex constraints. The traditional approaches to (1.5)–(1.8) obtain a solution by minimizing a measure of the misfit between $\boldsymbol{y}$ and $G^t\boldsymbol{f}$, usually measured by either the ℓ_2 or the ℓ_1 norm in the range of G^t, combined with some way to take the convex (box) constraints into account. Our approach consists of minimizing a smooth convex function $\Psi(\boldsymbol{\xi}) : \mathcal{K} \to \mathbb{R}$, subject to the equation to solve as a linear constraint, that is, to solve:

$$\text{Find} \quad \boldsymbol{\xi}^* = argmin\{\Psi(\boldsymbol{\xi}) : \boldsymbol{\xi} \in \mathcal{K}, \text{ and } \boldsymbol{A\xi} = \boldsymbol{y}\}. \tag{2.1}$$

We choose the function $\Psi(\boldsymbol{x})$ in such a way that the box constraints are automatically satisfied. The function that we propose to use is

$$\Psi(\boldsymbol{\xi}) = \sum_{j=1}^{(m+n)} \frac{\xi_j - a_j}{D_j} \ln\left(\frac{\xi_j - a_j}{D_j}\right) + \frac{b_j - \xi_j}{D_j} \ln\left(\frac{b_j - \xi_j}{D_j}\right). \tag{2.2}$$

Since, in our approach, we regard the message as well as the noise as unknowns, it is important to keep in mind that the first m-components of ξ correspond to the message (or signal), and the last n to the noise.

The function Ψ happens to be the Lagrange-Fenchel dual of the function

$$M(\tau) = \sum_{j=1}^{(m+n)} \ln\left(e^{a_j\tau_j} + e^{b_j\tau_j}\right), \quad \boldsymbol{\tau} \in \mathbb{R}^{(m+n)}, \tag{2.3}$$

which is the logarithm of the Laplace transform of a measure that puts unit mass at each corner of $\mathcal{K}$. The functions Ψ and M are related by:

$$\begin{aligned} \Psi(\boldsymbol{\xi}) &= \sup\{\langle \boldsymbol{\xi}, \boldsymbol{\tau}\rangle - M(\tau) | \boldsymbol{\tau} \in \mathbb{R}\}, \quad \boldsymbol{\xi} \in \mathcal{K}. \\ M(\tau) &= \sup\{\langle \boldsymbol{\tau}, \boldsymbol{\xi}\rangle - \Psi(\boldsymbol{\xi}) | \boldsymbol{\xi} \in \mathcal{K}\}, \quad \boldsymbol{\tau} \in \mathbb{R}^{n+m}. \end{aligned} \tag{2.4}$$

It is simple to verify that both $M(\boldsymbol{\tau})$ and $\Psi(\boldsymbol{\xi})$ are convex. Not only that, Ψ is infinitely differentiable in the interior $int(\mathcal{K})$ of $\mathcal{K}$. For $\boldsymbol{\tau} \in \mathbb{R}^{n+m}$, the equation $\boldsymbol{\tau} = \nabla\Psi(\boldsymbol{\xi})$ has a unique solution $\boldsymbol{\xi}$ in the interior of $\mathcal{K}$. Not only that, the following identity holds:

$$\nabla_\tau M(\tau) = (\nabla_\xi \Psi)^{-1}(\tau), \quad \text{when} \quad \nabla_\xi \Psi(\boldsymbol{\xi}) = \boldsymbol{\tau}. \tag{2.5}$$

That is, the gradients are inverse functions of each other. The interested reader may consider [2] for details about convexity, duality, etc.

Theorem 2.1. *Let* $\Psi(\boldsymbol{\xi})$ *and* $M(\boldsymbol{\tau})$ *be related to each other as in* (2.4). *Suppose that* $\boldsymbol{A}^t\boldsymbol{\lambda} \in int(\mathcal{K})$ *for any* $\boldsymbol{\lambda} \in \mathbb{R}^n$. *Then, the solution to* (2.1) *is given by:*

$$\xi_j^* = \frac{a_j e^{a_j(\boldsymbol{A}^t\boldsymbol{\lambda}^*)_j} + b_j e^{b_j(\boldsymbol{A}^t\boldsymbol{\lambda}^*)_j}}{e^{a_j(\boldsymbol{A}^t\boldsymbol{\lambda}^*)_j} + e^{b_j(\boldsymbol{A}^t\boldsymbol{\lambda})_j}}, \qquad j = 1, ..., (n+m). \tag{2.6}$$

Here, $\boldsymbol{\lambda}^* \in \mathbb{R}^{n+m}$ *is the point at which* $\Sigma(\boldsymbol{\lambda}, \boldsymbol{y}) \equiv \langle \boldsymbol{\lambda}, \boldsymbol{r}_e \rangle - M(A\lambda)$ *achieves its maximum value. Also,*

$$\Psi(\boldsymbol{\lambda}^*) = \Sigma(\boldsymbol{\lambda}^*, \boldsymbol{y}). \tag{2.7}$$

Keep in mind that the first m components of $\boldsymbol{\xi}^*$ solve problem (2.1) and the last n are the estimated measurement error $\boldsymbol{\epsilon}^*$. Notice as well that, since $\Sigma(\boldsymbol{\lambda}, \boldsymbol{r}) = \langle \boldsymbol{\lambda}, \boldsymbol{r} \rangle - M(A\lambda)$ is a strictly convex, infinitely differentiable function, its maximizer occurs at $\boldsymbol{\lambda}^*$, satisfying

$$\nabla_{\lambda}\Sigma(\boldsymbol{\lambda}, \boldsymbol{r}) = 0 \quad \Leftrightarrow \quad \boldsymbol{A}\boldsymbol{\xi}^* = \boldsymbol{y},$$

with $\boldsymbol{\xi}^*$ given by (2.6). Also, most numerical software packages [*] are written to solve a minimization problem by default; thus, instead of maximizing $\Sigma(\boldsymbol{\lambda}, \boldsymbol{y})$, it is convenient to minimize $-\Sigma(\boldsymbol{\lambda}, \boldsymbol{y})$.

2.1. The reconstruction error

When one solves problems (1.5) and (1.6) numerically, the solution $\boldsymbol{\xi}^*$ need not satisfy $\boldsymbol{A}\boldsymbol{\xi}^* = \boldsymbol{y}$ exactly. The reconstruction error just measures how large is the offset relative to the problem data. In methods that use $\|\boldsymbol{A}\boldsymbol{\xi} - \boldsymbol{y}\|^2$, the value of the objective function at the optimum, namely $\|\boldsymbol{A}\boldsymbol{\xi}^* - \boldsymbol{y}\|^2$, is also a measure of the reconstruction error. In our approach, the minimum value $\Psi(x^*)$ does not measure the quality of the reconstruction error. Nevertheless, we know from Theorem 2.1 that:

$$\|\nabla_{\lambda}\Sigma(\boldsymbol{\lambda}, \boldsymbol{y})\| = \|\boldsymbol{A}\boldsymbol{\xi}^* - \boldsymbol{y}\|. \tag{2.8}$$

reaches its smallest value at $\boldsymbol{\xi}^*$. If this value is zero, it means that the constraint is satisfied exactly. The size of the error is used as a halting criterion for the iterative minimization procedure. Once the norm of (2.8) is smaller than a preassigned amount 10^{-5} in our case, the algorithm stops.

3. Numerical examples

The four examples that follow correspond to the two possible interpretations of the additive error term. In the first two, the decoder receives the coded message, the last element of which is the sum of the confounding errors added to the coded message, that is, the ℓ_1-norm of the confounding noise.

For the third example, we start with an initial value of the ℓ_1-norm of the confounding (just to minimize the sweeping over possible values of the norm), and we decrease the norm by a factor 0.9 during each run of the optimization algorithm. We observe that as the ℓ_1-norm of the error gets smaller,

[*] See `https://metrumresearchgroup.github.io/bbr/` for example, which combines the usual gradient method with a step reduction procedure at each iteration. This is convenient because the objective function may be very flat near the minimum

the quality of the decoded message gets worse. This fact supports the interpretation of the error term as a slack variable to absorb the misfit between transmitted and received signals.

The fourth experiment is just a test of the reconstruction algorithm. We generate random code matrices, messages, and errors, and then test the algorithm within the framework of the first interpretation of the noise term. We observe that in some cases, the algorithm performed as in the first example.

3.1. Numerical example using a Reed-Solomon code matrix in $\mathbb{F}_{11}$

As homage to the number theoretic/algebraic approach, we use the Reed Solomon (RS) methodology to generate the input data for the example. However, the computations are carried out as ordinary real numbers. We consider linear codes of length n, dimension m, and minimum distance $d = n - m + 1$. The entries are from a finite field $\mathbb{F}_q$, where q is a prime (we used $q = 11$), with $q > n$. Let P_m be the set of polynomials in $\mathbb{F}_q[x]$ of degree less or equal to m. Integrating ILD and ELD,

Definition 3.1. *Let $x_1, x_2, \ldots, x_n$ be different elements of $\mathbb{F}_q$, then the codewords of the Reed Solomon code are $(p(x_1), p(x_2), \ldots, p(x_n))$ for all $p \in P_m$.*

The following matrix $\boldsymbol{G}_0$ is a generator matrix of the RS code:

$$\boldsymbol{G}_0 = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ x_1 & x_2 & \ldots & x_n \\ x_1^2 & x_2^2 & \ldots & x_n^2 \\ \vdots & & & \\ x_1^m & x_2^m & \ldots & x_n^m \end{pmatrix}. \tag{3.1}$$

In number theoretic/algebraic coding theory, there are decoding algorithms to solve (1.1), that is $\boldsymbol{y}^t = \boldsymbol{f}^t\boldsymbol{G} + \boldsymbol{e} \mod q$, where $\boldsymbol{f}$ is the vector that has the information that we want to recover, and $\boldsymbol{e}$ has Hamming weight less than $t = \lfloor \frac{n-m}{2} \rfloor$. We choose this as a first example to show that our proposal applies in the variants of McEliece using RS codes.

Let us define $\boldsymbol{G} = S\boldsymbol{G}_0P$, where S is an invertible matrix and P is a permutation matrix. In this case, the vector $\boldsymbol{e}$ is supposed to be known by the sender of the message but it is not known by the decoder, as he only knows t, the maximum Hamming weight of $\boldsymbol{e}$. We consider a problem with $m = 5$ and $n = 10$, therefore $m + n = 15$. The matrix $\boldsymbol{G}$ is given by

$$\boldsymbol{G} = \begin{pmatrix} 9 & 6 & 5 & 2 & 10 & 8 & 1 & 0 & 7 & 3 \\ 4 & 4 & 8 & 7 & 6 & 7 & 2 & 10 & 1 & 10 \\ 2 & 8 & 9 & 4 & 7 & 5 & 10 & 8 & 2 & 0 \\ 7 & 0 & 2 & 6 & 7 & 10 & 4 & 4 & 2 & 8 \\ 8 & 0 & 5 & 0 & 5 & 1 & 10 & 3 & 3 & 2 \end{pmatrix}. \tag{3.2}$$

The rest of the inputs are the following. The message and the added error, unknown to the decoder,

used by the sender are

$$\begin{aligned} \boldsymbol{f} &= (0, 6, 8, 7, 3), \\ \boldsymbol{e} &= (3, 0, 0, 0, 0, 0, 0, 0, 3, 0), \\ \boldsymbol{y} &= (3, 0, 6, 6, 2, 1, 7, 7, 1, 1), \\ s &= \boldsymbol{u}^t \boldsymbol{e} = 6, \\ \boldsymbol{y}_e &= (3, 0, 6, 6, 2, 1, 7, 7, 1, 1, 6). \end{aligned} \tag{3.3}$$

Even though the matrix is computed in modular arithmetics, the computation of $\boldsymbol{y}$ is not modular, to be in the same scenario as Candes and Tao [4].

Here, $\boldsymbol{f}$ is the signal transmitted and $\boldsymbol{e}$ is the confounding noise added by the sender (therefore we know s). The signal received by the decoder is $\boldsymbol{y}$. These are related among themselves as in (1.1), and for our purposes, as in (1.5) and (1.6). The job of the decoder is to use the methodology that we propose to recover $\boldsymbol{f}$ and $\boldsymbol{e}$ from the knowledge of $\boldsymbol{y}_e$ and $\boldsymbol{G}$.
For this first example, we choose $a_j = -10^{-3}$ and $b_j = 0.0900001$ for $j = 1, ..., 15$. The reason for this is that the signal and error are integers in the range $[0, 9]$, and before running the methodology, we divide the data (and therefore the observed values) by 100 to not have too large numbers in the exponents appearing in $M(\boldsymbol{A}\boldsymbol{\lambda})$.

To find the Lagrange multipliers $\boldsymbol{\lambda}^*$, we minimize $M(A\lambda) - \langle \boldsymbol{\lambda}, \boldsymbol{y} \rangle$ using the *BB* software in the R-library, specifying the gradient of the function, which happens to be

$$\sum_{j=1}^{15} (A_e)_{i,j} \frac{a_j e^{a_j (\boldsymbol{A}_e^t \boldsymbol{\lambda})_j} + b_j e^{b_j (\boldsymbol{A}_e^t \boldsymbol{\lambda})_j}}{e^{a_j (\boldsymbol{A}_e^t \boldsymbol{\lambda})_j} + e^{b_j (\boldsymbol{A}_e^t \boldsymbol{\lambda})_j}} - (y_e)_i, \quad i = 1, ..., 11.$$

Once the $\boldsymbol{\lambda}^*$ that makes this gradient zero is found, it is inserted in the representation (2.6). Then, we multiply the solution found by 100 and round to the nearest integer to recover the signal and the noise. Next, we present the result listed in Table 1 in three ways: First as the output of the computation, next as the same result multiplied by 100 and rounded off a bit, and finally the result rounded up to the nearest integer. We mention that the reconstruction error, calculated as indicated in (2.8) is 3.348618×10^{-5}.

The data and confounding errors are recovered correctly. We attempt the cases of $\mathbb{F}_{17}$ and $\mathbb{F}_{31}$ and recover the message and the error.

We choose Reed Solomon codes, since they are commonly used, for example, in mobile and wireless communication, CD, DVD, bar-codes, satellite communication, and high-speed modems, to mention some of them. In the cryptography case, we know that this variant has been considered; nevertheless, here we do it from another point of view, and we are not using the structure of the code. This gives us an interesting future work, trying to attack McEliece with Goopa codes.

Table 1. Results, rescaled and true values.

1	8.753008×10^{-6}	0.000875	0
2	6.000378×10^{-2}	6.000378	6
3	7.998502×10^{-2}	7.998502	8
4	6.998870×10^{-2}	6.998870	7
5	3.001820×10^{-2}	3.001820	3
6	2.986098×10^{-2}	2.986098	3
7	4.654880×10^{-5}	0.004654	0
8	-9.999731×10^{-6}	-0.000999	0
9	7.844249×10^{-5}	0.0078442	0
10	-1.000000×10^{-5}	-0.00100	0
11	6.498100×10^{-5}	0.006498	0
12	-9.967433×10^{-6}	-0.00099	0
13	6.763395×10^{-5}	0.006763	0
14	2.992630×10^{-2}	2.992630	3
15	-9.999991×10^{-6}	-0.00099	0

3.2. Numerical example using a Reed-Solomon code matrix in $\mathbb{F}_{2^5}$

For the second example, we consider $q = 2^5$ because this is a more realistic case in cryptography. We use the vector used to generate the RS-matrix as in (3.1), being $(1, 2, 3, ...30, 31)$. The generator matrix is a 15×31-matrix, and the code can correct up to 8 errors. Upon running the algorithm, all entries of the message are exactly reproduced, except sometimes that we do not recover 1 position. This is regardless of the number of non-zero components of the error vector.

It is interesting to remark that since we do not use the structure of the generator matrix of the code, even though we do not use modular arithmetic, this might be of interest in cryptoanalysis.

3.3. Minimizing the ℓ_1*-norm of the noise*

The thrust of this example is to show the effect of minimizing the ℓ_1 norm of the noise on the recovery of the transmitted signal, that is, on the recovery of the message. Here, we use the value $s = \langle \boldsymbol{u}, be \rangle$ introduced as a constraint in (1.7) and (1.8), as a control parameter. We start from a value slightly larger than the one in the dataset (3.3) and decrease it stepwise, and at each step, we solve (1.8) and (1.9) for the current value of s, and recover the message and the noise in the observed signal and record the error. This is done until the procedure stops because the error is too large. This exercise is related to the proposal in [4], except that here we do not minimize the ℓ_1-norm of the misfit between observation $\boldsymbol{y}$ and true signal $\boldsymbol{A}\xi$, instead, we minimize the entropy $\Psi(\boldsymbol{\xi})$ to recover the message $\boldsymbol{f}$ and the noise $\boldsymbol{e}$ in one shot.

As expected, since the current value of s decreases, the recovered noise will differ at each iteration, and be different from the initial noise that is added to the signal. In the numerical experiments, the recovery of the signal and the noise are correct only when the value of s used as a constraint is slightly larger than the original noise added to confound. The reason is that a large value of the ℓ_1-norm constraint, may lead to more error in the recovered message. As mentioned, this is due to the role of the estimated noise as a slack variable to compensate for the misfit between the observed message and

the true transmitted message. When the value of s as the control parameter is smaller than the actual norm of the noise present in the signal, the reconstruction error computed by (2.8) is larger.

To illustrate, consider the same data set as that in the previous example, namely, as in (3.3), but this time we consider a sequence of similar problems, one for each value of s. We consider the following sequence $s_n = (0.9)^{n-2}(0.06)$, with $s = 0.06$ as in the previous example and $n = 0, 1, ..., 10$. Note that we start with an initial value (thought of as an initial guess by the decoder) larger than the true datum ($s = 6/100$) of the previous example. Recall that to avoid numerical overflow, we divide the data by 100. This idea was originally developed as an entropic approach to linear programming in [9]. The results are summarized in Tables 2–4.

Table 2. Coincidences between decoded messages and data as $s = \|\boldsymbol{e}\|_{\ell_1}$ decreases.

n	$T = 1; F = 0$	s_n	∇
0	1	0.074	4.40×10^{-8}
1	1	0.066	$5.17 \times 10{-}8$
2	1	0.060	$8.84 \times 10{-}6$
3	1	0.055	0.001976
4	1	0.048	0.003781
5	1	0.043	0.005405
6	1	0.039	0.006868
7	1	0.035	0.008184
8	1	0.031	0.009368
9	1	0.028	0.010434
10	1	0.025	0.011393

In Tables 2 and 3, the first column is labeled by the power specifying the s_n, The second column header is $T = 1$ (standing for $TRUE = 1$) to mean that the reconstructed code coincides fully with the given trial code, or $F = 0$ (standing for $FALSE = 0$) to mean that there are components in the recovered code that do not coincide with the sent message. In both tables, s_n denotes the values of the norm $\|\boldsymbol{e}\|_{\ell_1}$ used as input, and ∇ stands for the reconstruction error, computed as explained in Section 2.1, and lists up to six significant figures. In all cases, it is of the order of 10^{-6} or smaller.

Next, we reconstruct only the error vector and have the following table:

Table 3. Coincidences between reconstructed and simulated error as $s = \|\boldsymbol{e}\|_{\ell_1}$ decreases.

n	$T = 1; F = 0$	s_n	∇
0	1	0.074	4.40×10^{-8}
1	1	0.066	$5.17 \times 10{-}8$
2	1	0.060	$8.84 \times 10{-}6$
3	0	0.055	0.001976
4	0	0.048	0.003781
5	0	0.043	0.005405
6	0	0.039	0.006868
7	0	0.035	0.008184
8	0	0.031	0.009368
9	0	0.028	0.010434
10	0	0.025	0.011393

Table 4. Errors: Total, in the code and in the noise.

n	s_n	Total	Code	Noise
0	0.074	4.40×10^{-8}	$4.27 \times 10{-}8$	1.05×10^{-8}
1	0.066	$5.17 \times 10{-}8$	$4.98 \times 10{-}8$	1.37×10^{-8}
2	0.060	9.30×10^{-6}	$8.84 \times 10{-}6$	2.89×10^{-6}
3	0.054	0.002116	0.001976	0.000758
4	0.048	0.004049	0.003781	0.001450
5	0.043	0.005789	0.005405	0.002072
6	0.039	0.007355	0.006868	0.002633
7	0.035	0.008765	0.008184	0.003138
8	0.031	0.010033	0.009368	0.003592
9	0.028	0.011174	0.010434	0.003592
10	0.025	0.012202	0.011393	0.004368

Notice that when reconstructing the vector of errors, there is a success only in three cases, that is, when the norm of $\boldsymbol{e}$ is slightly larger or equal to the norm of the error added to the message as a confounding error.

3.4. Randomly generated code matrices

We apply this method to the random Gaussian matrices, and we are able to correct the errors. For a given n, the more the rate is closer to 0,6 the less we can correct errors. The same happens with q, for the same parameters, if $q_1 < q_2$, we can correct more errors using q_1 elements than q_2. Our motivation to decode these codes come from the paper by Candes and Tao [4], and we propose another method to solve it. Additionally, this shows us that the method is not using the structure of the matrix, as in the previous example, and is not using the structure of $\boldsymbol{G}$. This can be used in cryptanalysis of post-quantum code-based cryptographic primitives.

4. Conclusions

To sum up, the major features of our approach consist of: The definition of a strictly convex function whose domain is the constraint set, and such that its minimization yields an explicit representation of the solution to problems (1.4) or its extended version (1.7). The extension can be thought of as a regularization of the original problem to take care of the case in which the data does not belong to the range of the operator.

One important fact that shows up clearly in our numerical experiments is the potential use of confounding errors to spoil the norm minimization algorithms. We see that when there is a confounding error added to the message, the norm minimization algorithm does not yield a good reconstruction of the signal.

Also, we point out the potential applicability of the entropy minimization approach to post-quantum code-based cryptoanalysis, in particular, our approach can be tried on McEliece cryptosystems variants.

We also mention the potential applicability of the method to noisy Fourier inversion with scarce data.

Author contributions

All authors contributed equally to this work. All authors have read and approved the final version of the manuscript for publication.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The authors would like to thank the reviewers and the editors for their suggestions for improvements.

All authors acknowledge financial provided by the Vice Presidency for Research & Creation publication fund at the Universidad de los Andes.

Conflict of interest

The authors declare no conflict of interest.

References

1. F. L. Bauer, *Decrypted secrets: Methods and maxims on cryptography*, Berlin: Springer-Verlag, 1997.

2. J. M. Borwein, A. S. Lewis, *Convex analysis and nonlinear optimization*, 2nd Edition, Berlin: CMS-Springer, 2006.
3. D. Burshtein, I. Goldenberg, Improved linear programming decoding and bounds on the minimum distance of LDPC codes, *IEEE Inf. Theory Work.*, 2010. Available from: `https://ieeexplore.ieee.org/document/5592887`.
4. E. Candes, T. Tao, *Decoding by linear programming*, *IEEE Tran. Inf. Theory*, **51** (2005), 4203–4215. http://dx.doi.org/10.1109/TIT.2005.858979
5. E. Candes, T. Tao, Near optimal signal recovery from random projections: Universal encoding strategies, *IEEE Tran. Inf. Theory*, **52** (2006), 5406–5425. http://dx.doi.org/10.1109/TIT.2006.885507
6. C. Daskalakis, G. Alexandros, A. G. Dimakis, R. M. Karp, M. J. Wainwright, Probabilistic analysis of linear programming decoding, *IEEE Tran. Inf. Theory*, **54** (2008), 3565–3578. http://dx.doi.org/10.1109/TIT.2008.926452
7. S. El Rouayyheb, C. N. Georghiades, Graph theoretic methods in coding theory, *Classical, Semi-class. Quant. Noise*, 2012, 53–62. https://doi.org/10.1007/978-1-4419-6624-7_5
8. J. Feldman, M. J. Wainwright, D. R. Karger, Using linear programming to decode binary linear codes, *IEEE Tran. Inf. Theory*, **51** (2005), 954–972. https://doi.org/10.1109/TIT.2004.842696
9. F. Gamboa, H. Gzyl, Linear programming with maximum entropy, *Math. Comput. Modeling*, **13** (1990), 49–52.
10. Y. S. Han, A new treatment of priority-first search maximum-likelihood soft-decision decoding of linear block codes, *IEEE Tran. Inf. Theory*, **44** (1998), 3091–3096. https://doi.org/10.1109/18.737538
11. M. Helmiling, Advances in mathematical programming-based error-correction decoding, *OPUS Koblen.*, 2015. Available from: `https://kola.opus.hbz-nrw.de/frontdoor/index/index/year/2015/docId/948`.
12. M. Helmling, S. Ruzika, A. Tanatmis, Mathematical programming decoding of binary linear codes: Theory and algorithms, *IEEE Tran. Inf. Theory*, **58** (2012), 4753–4769. https://doi.org/10.1109/TIT.2012.2191697
13. M. R. Islam, Linear programming decoding: The ultimate decoding technique for low density parity check codes, *Radioel. Commun. Syst.*, **56** (2013), 57–72. https://doi.org/10.3103/S0735272713020015
14. T. Kaneko, T. Nishijima, S. Hirasawa, An improvement of soft-decision maximum-likelihood decoding algorithm using hard-decision bounded-distance decoding, *IEEE Tran. Inf. Theory*, **43** (1997), 1314–1319. https://doi.org/10.1109/18.605601
15. S. B. McGrayne, *The theory that would not die. How Bayes' rule cracked the enigma code, hunted down Russian submarines, & emerged triumphant from two centuries of controversy*, New Haven: Yale University Press, 2011.
16. R. J. McEliece, A public-key cryptosystem based on algebraic, *Coding Th.*, **4244** (1978), 114–116.
17. H. Mohammad, N. Taghavi, P. H. Siegel, Adaptive methods for linear programming decoding, *IEEE Tran. Inf. Theory*, **54** (2008), 5396–5410. https://doi.org/10.1109/TIT.2008.2006384

18. G. Xie, F. Fu, H. Li, W. Du, Y. Zhong, L. Wang, et al, A gradient-enhanced physics-informed neural networks method for the wave equation, *Eng. Anal. Bound. Ele.*, **166** (2024). https://doi.org/10.1016/j.enganabound.2024.105802

19. Q. Yin, X. B. Shu, Y. Guo, Z. Y. Wang, Optimal control of stochastic differential equations with random impulses and the Hamilton-Jacobi-Bellman equation, *Optimal Control Appl. Methods*, **45** (2024), 2113–2135. https://doi.org/10.1002/oca.3139

20. B. Zolfaghani, K. Bibak, T. Koshiba, The odyssey of entropy: Cryptography, *Entropy*, **24** (2022), 266–292. https://doi.org/10.3390/e24020266

Appendix: A complementary computation

To conclude we present a small computation that explains why we augmented the sizes of the box constraints to avoid numerical overflow when the solutions get close to the boundary of the set. Since the objective function and its Fenchel-Lagrange dual are separable (that is, sums of functions of one variable), it suffices to consider a one-dimensional situation. Let $m(\tau) = \ln(e^{a\tau} + e^{b\tau})$. The Fenchel-Lagrange dual of this function is $\psi(\xi) = max\{\xi\tau - m(\tau)|\tau \in \mathbb{R}\}$. The first order condition for $\boldsymbol{\xi}$ to be a maximizer is that

$$\xi = \frac{ae^{a\tau} + be^{b\tau}}{e^{a\tau} + e^{b\tau}} \quad \Leftrightarrow \quad \tau = \left(\frac{b-\xi}{\xi-a}\right)^{1/(b-a)}.$$

On the one hand, substituting in the definition of ψ we obtain

$$\psi(\xi) = \frac{b-\xi}{b-a}\ln\left(\frac{b-\xi}{b-a}\right) + \frac{\xi-a}{b-a}\ln\left(\frac{\xi-a}{b-a}\right).$$

And, on the other hand, observe that as $\tau \to \pm\infty$, then $\xi \to b$ or $\xi \to a$ respectively. So, the Fermi-Dirac entropy considered is supported by $[a, b]$, and the dual optimization problem is unconstrained.