International Journal of Electrical and Computer Engineering (IJECE)
Vol. 15, No. 6, December 2025, pp. 5411~5421
ISSN: 2088-8708, DOI: 10.11591/ijece.v15i6.pp5411-5421

Optimal design, decoding, and minimum distance analysis of Goppa codes using heuristic method

Bouchaib Aylaj[1], Said Nouh[2], Mostafa Belkasmi[3]
[1]Department of Computer Science, Regional Center for Professions of Education and Training (CRMEF), Rabat, Morocco
[2]Faculty of Sciences Ben M'Sick, Hassan 2 University, Casablanca, Morocco
[3]National Higher School of Computer Science and Systems Analysis (ENSIAS), Mohamed V University, Rabat, Morocco

Article Info

Article history:

Received Dec 30, 2024
Revised Jul 25, 2025
Accepted Sep 15, 2025

Keywords:

Communication system
Construction
Decoding
Goppa codes
Heuristic methods
Minimum distance

ABSTRACT

Error-correcting codes are crucial to ensure data reliability in communication systems often affected by transmission noise. Building on previous successful applications of our heuristic method degenerate quantum simulated annealing (DQSA) to Bose–Chaudhuri–Hocquenghem (BCH) and quadratic residue (QR) codes. This paper proposes two algorithms designed to address two coding problems for Goppa codes. DQSA-d_{min} computes the minimum distance (d_{min}) while DQSA-Dec, serves as a hard decoder optimized for additive white gaussian noise (AWGN) channels. We validate DQSA-d_{min} comparing its computed minimum distances with theoretical estimates for algebraically constructed Goppa codes, showing accuracy and efficiency. DQSA-d_{min} further used to find the optimal Goppa codes that reach the lower bound of d_{min} for linear codes known in the literature and stored in Marcus Grassl's online database. Indeed, we discovered 12 Goppa codes reaching this lower bound. For DQSA-Dec, experimental results show that it obtains a bit error rate (BER) of 10-5 when SNR=7.5 for codes with lengths less than 65, which is very interesting for a hard decoder. Additionally, a comparison with the Paterson algebraic decoder specific to this code family shows that DQSA-Dec outperforms it with a 0.6 dB coding gain at BER=10-4. These findings highlight the effectiveness of DQSA-based algorithms in designing and decoding Goppa codes.

Corresponding Author:

Bouchaib Aylaj
Department of Computer Science, Regional Center for Professions of Education and Training (CRMEF)
10000 Av. Allal Al Fassi, Rabat 11000, Morocco
Email: bouchaib_aylaj@yahoo.fr

1. INTRODUCTION

Error-correcting codes are at the heart of modern communication systems, contributing essentially to maintaining the integrity of data transmitted over noisy and disrupted channels. They enable the detection and correction of errors occurring during data transmission, particularly in environments with high variability such as in communication systems as shown in Figure 1. In particular mobile systems [1], [2], where signals can be affected by interference, noise or losses due to unstable transmission conditions.

The optimal and efficient choosing or designing of correction codes in communication systems represents a fundamental challenge, as it conditions the ability to identify and correct transmission errors. Criteria such as the minimum distance (d_{min}), which reflects the correction capability, the encoding rate, the simplicity of encoding, and the efficiency of the decoders, are critical to ensuring reliable and efficient transmission [3], [4]. However, computing the d_{min} and decoding the codes are known to be NP-difficult problems [5], [6]. Among the proposed solutions, Goppa's codes stand out for their excellent structural

properties and robustness [7], [8], which makes them particularly suitable in communication systems [9] and cryptography [10]. These codes offer good error correction thanks to algorithms such as the Patterson decoder [11] and are also studied in the context of Gaussian noise-resilient systems (AWGNs). However, despite their advantages, optimizing their design, efficiently computing their d_{min} and developing performance decoders remain open issues, requiring innovative approaches to complement traditional algebraic methods. The study of Goppa codes has traditionally relied on algebraic methods for the processing of key tasks such as computing the minimum distance, decoding, and constructing these codes. These approaches often exploit the mathematical structure of Goppa codes. However, despite their efficiency, algebraic techniques can sometimes be computationally expensive or limited, especially for large codes. According to our literature review and to our knowledge, almost no studies have explored the use of heuristic methods to solve these problems for Goppa codes, thus leaving a significant gap in the exploration of alternative approaches.

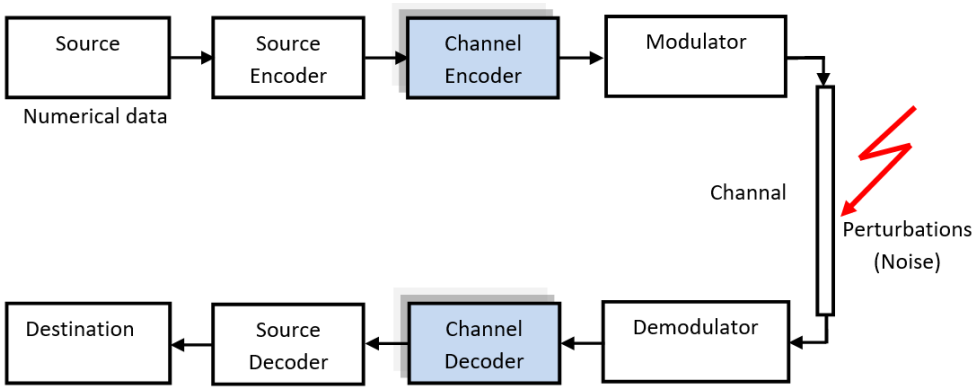

Figure 1. The general system of communication

Several notable works have advanced the development of algebraic techniques in this domain. For instance, the Patterson decoder [11] and Berlekamp-Massey algorithm [12] remain an essential reference for the efficient decoding of Goppa codes. Similarly, algebraic constructions [13]–[15] have widely studied to generate codes with interesting parameters by exploiting the properties of Goppa codes such as polynomials and finite elements. Research on minimum distance estimation [16]–[19] often relies on combinatorial or algebraic bounds which, while rigorous, can be difficult to compute directly for complex codes. Faced with this predominance of algebraic methods, it seems innovative and promising to explore heuristic methods as a complementary tool. Heuristic methods, such as simulated annealing offer flexibility and the ability to address complex optimization tasks that might be difficult for purely algebraic methods. Simulated annealing is an optimization technique that draws inspiration from the slow cooling of materials in metallurgy, first presented by Černý in 1985 [20] and Kirkpatrick *et al.* in 1983 [21]. With the use of this method, we aim to fill this gap to address the problems of minimum distance calculation, decoding and construction of optimal Goppa codes.

In this paper, we extend the degenerate quantum simulated annealing (DQSA) method, which has been successfully tested on BCH and QR codes in previous works [22]–[24], to address both of these problems for Goppa codes. Two proposed algorithms have been developed:

- DQSA-d_{min} calculator designed to compute d_{min} quickly and accurately.
- DQSA-Dec: a hard decoder optimized for AWGN channels tested and compared to algebraic Patterson decoder.

We also used DQSA-d_{min} to validate the quality of algebraically constructed Goppa codes by comparing their calculated minimum distances with the theoretical limits. Finally, DQSA-d_{min} identified 12 codes reaching the theoretical lower bound of d_{min} for linear codes, validated via the Marcus Grassl database [25] and Brouwer's tables [26].

The paper begins with section 2 which provides an overview of error-correcting codes, in particular Goppa codes. Section 3 introduces our DQSA heuristic method used for the problems related to minimum distance and decoding Goppa codes. Section 4 details the DQSA-d_{min} calculator, its algorithm, and its efficiency in calculating minimum distances as well as identifying optimal codes. Section 5 focuses on the DQSA-Dec decoder, outlining its superior decoding algorithm and performance. Finally, section 6 summarizes the main contributions and proposes to extend DQSA to other code families and new applications. In this paper, several abbreviations and symbols are used for concepts. The Table 1 contains their meanings.

Table 1. Signification of symbols and abbreviations

Symbol or abbreviation	Signification	Symbol or abbreviation	Signification
DQSA	Degenerate quantum simulated annealing method	SNR	Signal-to-noise ratio
BCH	Bose–Chaudhuri–Hocquenghem codes	dB	Decibel
QR	Quadratic residues codes	C *(n, k, d_{min}) or C*	Linear code
d_{min}	Minimum distance of a code	GC*(n, k, d_{min}) or GC*	Goppa code
DQSA-d_{min}	Degenerate Quantum simulated annealing to determine a code's minimum distance	*n*	Code length (number of bits per codeword)
DQSA-Dec	Degenerate Quantum simulated annealing algorithm a Hard decoder of a Goppa code	*k*	Code dimension (number of information bits).
AWGN	Additive white Gaussian noise	k/n	Coding ratio of code
BER	Bit error rate, measuring the proportion of errors in the received bits	tc	Error correcting capability of code
GF(q)	Finite field of size ***q***	$H = (h_{ij})_{n-k\times n}$	Parity-check matrix of the code
m	Degree of the finite field extension	S*(RV)*	Syndrome of the received vector ***RV***
g	Goppa polynomial of degree r	Primary search system (PSS)	PSS
α	Primitive element of a finite field in GF(2^m)	Equivalence search system (ESS)	ESS
L	Set of points defining the support of the Goppa polynomial.	*E*	Function to evaluate Hamming weight of a code word
$W_H(V)$	Hamming weight of a codeword V	*T*	Temperature is a control parameter of DQSA
d_H	Hamming distance	T_i, T_f	Initial temperature, final temperature
$U=(u_1, \ldots, u_k)$	Information vector	δ	Rate of temperature reduction in DQSA
$G = (g_{ij})_{k\times n}$	Generator matrix of the code	Patterson dec	An algebraic decoding algorithm specifically designed for Goppa codes
$C^{\perp}(n, n-k)$	Dual code	CodeTable	An online database containing parameters of optimal linear codes [25]

2. ERRORS CORRECTING CODES

2.1. Goppa codes

Error-correcting codes are divided into two main families: linear and non-linear codes. Common types include block codes (where data is broken into blocks) and convolutional codes (where information is processed in streams). Goppa codes are linear codes that form a type of error-correcting codes defined from polynomials and algebraic curves. Valerii Goppa [7] invented Goppa codes in 1970. They were first studied for their properties as error-correcting codes, and then, with the appearance of the *MCELIECE* cryptosystem, they were studied for their cryptographic properties.

Definition:

A Goppa code is built on a finite field $GF(q^m)$ where $m \geq 1$ is an integer and q is a power of a prime number $n < q^m$. Let g a polynomial of degree r, $g \in GF(q^m)\ [x]$ and $L = \{\alpha_1, \ldots, \alpha_n\} \subset GF(q^m)$. The α_i are two distinct by two. $(card(L) = n)$, $g(\alpha_i) \neq 0$ for all $i = 1, \ldots, n$. The Goppa code denoted $\Gamma(L, g)$ is:

$$\Gamma(L, g) = \left\{ w = (w_1, \ldots, w_n) \epsilon\, GF(q^m) / \sum_{i=1}^{n} \frac{w_i}{x - \alpha_i} \equiv 0\ mod\ g(z) \right\} \quad (1)$$

$g(z)$ is called the Goppa polynomial; and $\Gamma(L, g)$ is a linear code of length n, dimension $k \geq n - mr$ and minimum distance $d \geq r + 1$.

2.2. Linear binary block codes

In this paper, we concentrate on linear binary block codes where the $q = 2$. Consider a block code $C(n, k, d_{min})$. Each member $V \epsilon C$ is referred to as a codeword, there are 2^k codewords in total, forming a k-dimensional subspace of the vector space $GF(2)^n$. When the modulo-2 sum of any two codewords is also a codeword, the code *C* is referred to as linear. For a codeword *V*, the number of nonzero components is the Hamming weight, or $W_H(V)$. Two codewords V_1 and V_2 differ in the number of locations they occupy, which is known as the Hamming distance, or $d_H\ (V_1, V_2)$. The lowest distance between any two different codewords in the code is known as the minimum Hamming distance or the minimum distance (d_{min}) of code *C*.

$$dmin = d_H(V_i, V_j) \quad \forall\, V_i, V_j \in C \quad (2)$$

It can be easily shown that the Hamming distance between two codewords in a linear block code *C* is equal to the Hamming weight of the modulo-2 sum ($\oplus$) of the two codewords, as expressed,

$$dmin = W_H(V_i \oplus V_j) \quad \forall V_i, V_j \in C \tag{3}$$

The code C's generator is G denoted $G = (g_{ij})_{k\times n}$ where its rows form a collection of basis vectors for the subspace $GF(2)^k$. A unique representation of each codeword $V = (v_1, \dots, v_n)$ can then be obtained by combining the rows of G in a linear form.

$$\forall \ 1 \le j \le n \qquad v_j = \overset{i=1}{\overset{k}{\oplus}} u_i g_{ij} \tag{4}$$

Where $U = (u_1, \dots, u_k) \in \{0,1\}^k$ information vector.

$$V' \in C^{\perp} \Leftrightarrow \forall V \in C: \ V'.V = 0 \tag{5}$$

"." indicates the scalar product.
The code $C^{\perp}$ *(n, n – k)* which is defined by (5) is linear as well, known as the dual code of *C,* and its generator matrix is represented by $H = (h_{ij})_{n-k\times n}$ also referred to as the parity-check matrix. $S(RV) = HRV$ is the vector S that results from multiplying the received vector RV by the matrix H. The syndrome is the name of this vector S. When the received vector has errors, the syndrome will not be zero.

3. THE PROPOSED HEURISTIC METHOD

Unlike classical methods to simulated annealing, which rely on a single processing system. Our proposed heuristic method based on the simulated annealing algorithm, introduces two distinct subsystems, exploiting the properties of degenerate quantum systems, where several quantum states share the same energy. This leads to the creation of a new method called degenerate quantum simulated annealing (DQSA) [22]–[24]. The DQSA consists of two distinct processing subsystems: the PSS and the ESS:

a. Primary search system (PSS): This subsystem operates similarly to a traditional simulated annealing algorithm. It evolves a non-equivalent state using carefully chosen and varied neighbor functions to search for new neighboring states.
b. Equivalence search system (ESS): When the PSS encounters equivalent states, the ESS takes over. It explores alternative states with the same energy, generating and evaluating several equivalent states to find the most promising one.

The DQSA, is adapted to efficiently explore the solution space, offering a DQSA-d_{min} calculator algorithm capable of determining the minimum distance between code words. At the same time, our DQSA method allows us to develop a high-performance DQSA-Dec Hard decoder algorithm, optimizing the correction of errors in a received code word. As a heuristic method, the DQSA-based algorithms performances depend on several parameters. The initial values of the DQSA-d_{min} calculator in Algorithm 1 and the DQSA-Dec decoder Algorithm 3), *i.e.*, the initial T_i and final temperatures T_f, the cooling rate θ, the number of iterations *N* and the Starting subsystem, were optimized through 15 numerical tests. In each trial, these parameters were varied and combined to assess their influence on performance. After analyzing the results, the average of the best-performing configurations was selected as the optimal configuration.

4. DQSA-d_{min} CALCULATOR ALGORITHM

To show how DQSA applies to computing the minimum distance of Goppa codes, we present an analogy between the physical model of DQSA and its algorithmic (DQSA-d_{min} Calculator) use in optimization. Concepts like energy states are mapped to cost functions, enabling an efficient search for optimal codes. This relationship is detailed in algorithm 1 and summarized in Table 2.

4.1. Determination of the Function (E) to evaluate in algorithm 2 of DSA-d_{min} calculator

By substituting (2) into (3) and taking into account the fact that.

$$W_H(V) = \sum_{j=1}^{n} v_j \tag{6}$$

$$We\ have \quad dmin = \min_{\substack{u\in\{0,1\}^k \\ u\neq 0}} \sum_{j=1}^{n} \left(\overset{k}{\underset{i=1}{\oplus}} u_i g_{ij}\right) \tag{7}$$

$$Let \qquad E(V) = \sum_{j=1}^{n} \left(\overset{k}{\underset{i=1}{\oplus}} u_i g_{ij}\right) \tag{8}$$

$$where\ \ U = (u_1, \ldots, u_k) \in \{0,1\}^k - \ \{0\}$$

Thus, the function $E(V)$ returns the Hamming weight of the codeword V.

Table 2. The analogy between DQSA and DQSA-d_{min} calculator

DQSA method	DQSA-d_{min}lculator
PSS state	The information vector's codeword having a specific Hamming weight.
ESS state	The information vector's codeword having the same given Hamming weight.
Energy (E)	E= value of the Hamming weight of a given codeword
Neighbor state	Generating a new information vector having in the case of: 1. **PSS treatment**: a specific Hamming Weight 2. **ESS treatment**: the same Hamming Weight
Temperature	Controlling the calculator via iterations number
final state	Final result (codeword having the least Hamming weight)

4.2. DSA-d_{min} calculator algorithm

Algorithm 1 represents the steps of our DSA-d_{min} calculator.

Algorithm 1. DSA-d_{min} minimum Hamming weight calculator

```
Inputs:
 1. Tl_I: Total Iterations by temperature value, Tl_I ∈ [20, 5000]
 2. Ti=1.5, Tf=0.002, δ=0.89
 3. Starting subsystem= PSS
Output:
Value of codeword having the least Hamming weight
 1. While (T > Tf) do:
 2.    For iteration from 1 to Tl_I do:
 3.           If the current subsystem is PSS then Generate neighbor state (Ui+1) from PSS
      processing;
 4.           Else generate neighbor state (Ui+1) from ESS processing;
 5.           End if
 6.           Evaluate ΔE = E (Vi+1) - E (Vi);
 7.           If ΔE ≤ 0 then Ui ← Ui+1;
 8.           Else if (random (0, 1) ≤ Exp (-ΔE/T))
 9.                  Then Ui ← Ui+1;
 10.                   End if
 11.          End if
 12.    End For
 13.   With certain probability, switch between PSS and ESS;
 14.    T ← δ*T;
 15. End while
```

4.3. Algebraic construction

The Goppa codes used in this study were constructed algebraically, from specific polynomials $g(z)$ and *L-sets* points defined on a finite field. The minimum distance cannot be determined directly, it is estimated based on the error correction capability t, but This construction has been optimized to ensure minimum dimensions and distances close to the theoretical limits. In the Table 3 contains our construction of the Goppa codes on $GF(2^m)$ chosen to evaluate our DQSA-d_{min} calculator.

Table 3. Our construction of Goppa codes

Our construction of Goppa codes Goppa code parameters				Minimum distance	Code Goppa
Polynomial g(z)	Set of Points *L*	n	k	estimated	notation
z^2+z+1	$\{\alpha^i / i\ in\ [0\ldots20]\} \subset GF(2^5)$	21	11	5	$GC(21,11)$
z^4+z^3+1	$\{\alpha^i / i\ in\ [1\ldots31]\} \subset GF(2^5)$	31	11	9	$GC(31,11)$
z^6+z+1	$\{\alpha^i / i\ in\ [3\ldots64]\} \subset GF(2^7)$	62	20	13	$GC(62,20)$
$z^{10}+z^3+1$	$\{\alpha^i / i\ in\ [0\ldots117]\} \subset GF(2^7)$	117	47	21	$GC(117,47)$
z^7+z^5+1	$\{\alpha^i / i\ in\ [2\ldots127]\} \subset GF(2^7)$	126	77	15	$GC(126,77)$
z^6+z^3+1	$\{\alpha^i / i\ in\ [1\ldots195]\} \subset GF(2^8)$	195	147	13	$GC(195,147)$
$z^5+z^3+z^2+z+1$	$\{\alpha^i / i\ in\ [3\ldots219]\} \subset GF(2^8)$	217	177	11	$GC(217,177)$
$z^{13}+z^{10}+z^{19}+z+1$	$\{\alpha^i / i\ in\ [1\ldots255]\} \subset GF(2^8)$	255	151	27	$GC(255,151)$
$z^7+z^6+z^5+z^4+z^2+z+1$	$\{\alpha^i / i\ in\ [1\ldots255]\} \subset GF(2^8)$	255	199	15	$GC(255,199)$
$z^7+z^6+z^5+z^4+z^2+z+1$	$\{\alpha^i / i\ in\ [1\ldots305]\} \subset GF(2^9)$	305	242	15	$GC(306,242)$
$z^6+z^3+z^2+z+1$	$\{\alpha^i / i\ in\ [1\ldots315]\} \subset GF(2^9)$	315	261	13	$GC(315,270)$

4.4. Evaluation the DQSA-d_{min} to compute minimal distance of Goppa codes

The objective of the evaluation is to test the effectiveness of DQS-d_{min} on Goppa codes whose estimated minimum distance is calculated. For the Goppa codes constructed in this study, the calculated minimum distances were systematically compared to the theoretical estimates. In parallel, for each result found we calculate the number of iterations and the computation time needed to find such a value.

The results in Table 4 present a comparison between the calculated and estimated minimum distances of various Goppa codes using our DQSA-d_{min} Calculator. In general, the calculated distances align closely with the estimated ones, with minor discrepancies in some cases (*e.g.*, GC(217,177) shows 10 instead of 11, and GC(315,261) shows 14 instead of 13). The number of iterations varies significantly, with some codes requiring extensive iterations (*e.g.*, GC(195,147) with over 465,000 iterations), while others converge much faster (*e.g.*, GC(64,50) with just 345 iterations). Although the computation time is substantial for larger codes, it remains manageable, with the longest time being 184 seconds. These results indicate that our Calculator is effective, even as the code length methods 300 and the code rate nears 1/2, which increases computational complexity. Despite this, DQS-d_{min} successfully finds the estimated values.

Table 4. Results of DQSA-d_{min} for Goppa codes

GOPPA CODE	Minimum distance estimated theoretically	Minimum distance found by DQSA-d_{min} Calculator	Iteration number	Run Time(s)
$GC(21,11)$	5	5	80	< 1
$GC(31,11)$	9	9	108	< 1
$GC(62,20)$	13	13	305	< 1
$GC(117,47)$	21	21	119087	5
$GC(126,77)$	15	15	15867	< 1
$GC(195,147)$	13	13	465609	39
$GC(217,177)$	11	10	107345	10
$GC(255,151)$	27	29	1414569	184
$GC(255,199)$	15	15	433467	23
$GC(305,242)$	15	16	86193	6
$GC(315,261)$	13	14	145123	10

4.5. Finding the optimal Goppa codes using DQSA-d_{min}

Once evaluated and validated, DQSA-d_{min} was used to identify Goppa codes reaching the theoretical lower bound of d_{min} for linear codes existing in the literature. This search discovered 12 codes that matched this bound as reported in the Marcus Grassl database codes [25]. For this, as shown in Algorithm 2, we aim to find an optimal Goppa code by maximizing the d_{min} while respecting the code parameters. This starts with initialization, setting d_{min} to zero and constructing the *GF(2^m)* and *P(x)*. Using a specified number of iterations, it randomly generates L sets of *GF(2^m)* elements and a *g(z)* of the given degree. The algorithm ensures that *g(z)* is irreducible and that no element in L is the root of *g(z)*. It then uses the DQSA-d_{min} calculator function to calculate d_{min} for each found configuration of the code. If the resulting d_{min} is the greatest and equal to or greater than the theoretical lower bound, the corresponding parameters L, g(z), and code dimensions are stored as optimal. This process continues until the best configuration is identified.

Algorithm 2. To find an optimal Goppa code

Inputs:

1. n: where $n \leq 2^m$.
2. k: where $k < n$.
3. m: Degree of extension of the $GF(2^m)$.
4. *Lower_bd*: lower bound of d_{min} for linear codes existing in the literature
5. *degree_g*: Degree of the Goppa polynomial $g(z)$.
6. *num_iterations*: Number of iterations for the random search.
7. DQSA-d_{min} (n,k,L,g) calculator function: Returns d_{min} for the Goppa code defined by *n,k,L,g*.

Outputs:

1. Maximum of minimum distance d_{min}.
2. Optimal elements set L over *GF(2^m)*.
3. Optimal polynomial *g(z)*.
4. *k* (effective code dimension).
5. *n* (effective code length).

A. Initialize: $d_{min} \leftarrow 0$, $L \leftarrow \emptyset$, $g \leftarrow 1$, $k \leftarrow 0$
B. Construct: *GF(2^m)* and *P(x)*
C. **For** i from 1 to num_iterations **Do:**

```
    1. Randomly generate L = {α^j : j ∈ [1, n]} in GF(2^m)
    2. Generate a polynomial g(z) of degree degree_g
    3. If (g(z) is irreducible AND ∄ x ∈ L such that g(z) = 0) THEN
       • Calculer d_min ← DQSA-d_min(n, k, L, g)
       • If (d_min is maximum AND equal to lower_bd) THEN store d_min, L, g, and k
       • End If
    4. End If
D. End For
```

The Table 5 highlights the discovery of 12 Goppa codes with n between 100 and 106. These codes reach the lower limit of d_{min} for linear codes, the use of the generator polynomial rather than a generating matrix, makes the coding simplified, and the algebraic operations more efficient. With a coding rate close to 1/2, these codes offer an excellent balance between redundancy and efficiency. In addition, their error correction capacity, ranging from 4 to 7 errors, makes them ideal candidates for communication systems requiring reliability and robustness.

For each identified code, the parameters, including the set L, the Goppa polynomial $g(z)$, and other specifications, were input into the algebraic calculator of Magma [27]. This enabled an independent recalculation of the d_{min}, confirming the consistency and accuracy of the results obtained using the heuristic DQSA-d_{min} calculator, as shown in Table 5. This additional validation strengthens the reliability of our heuristic method.

Table 5. List of the 12 discovery Goppa code

No	Goppa Code Parameters				d_{min} found by DQSA	d_{min}'s Lower bound	d_{min} by Magma [27] Calculator	Advantages of the found code
	Polynomial $g(z)$	Set of Points L	n	k				
1	$z^4+z^3+z^2+z+1$	$\{\alpha^i/i\ in\ [1\ldots105]\} \subset GF(2^7)$	105	77	**9**	**9**	**9**	1. Reaching the lower bound of d_{min} for equivalent linear codes. 2. Coding based on a polynomial generator simplifying algebraic operations, instead of the generator matrix. 3. Coding rate close to 1/2. 4. Correcting capability between 4 and 7 errors.
2	$z^5+z^4+z^3+z^2+z+1$	$\{\alpha^i/i\ in\ [1\ldots105]\} \subset GF(2^7)$	105	70	**11**	**11**	**11**	
3	z^3+z^2+z	$\{\alpha^i/i\ in\ [1\ldots105]\} \subset GF(2^7)$	105	84	**7**	**7**	**7**	
4	$z^5+z^4+z^3+z^2+z$	$\{\alpha^i/i\ in\ [1\ldots106]\} \subset GF(2^7)$	106	71	**11**	**11**	**11**	
5	z^3+z^2+z+1	$\{\alpha^i/i\ in\ [1\ldots100]\} \subset GF(2^7)$	100	79	**7**	**7**	**7**	
6	$z^4+z^3+z^2+z+1$	$\{\alpha^i/i\ in\ [1\ldots100]\} \subset GF(2^7)$	100	72	**9**	**9**	**9**	
7	$z^5+z^4+z^3+z^2+z+1$	$\{\alpha^i/i\ in\ [1\ldots102]\} \subset GF(2^7)$	102	67	**11**	**11**	**11**	
8	$z^6+z^5+z^4+z^3+z^2+z+1$	$\{\alpha^i/i\ in\ [1\ldots102]\} \subset GF(2^7)$	102	60	**13**	**13**	**13**	
9	$z^7+z^6+z^5+z^4+z^3+z^2+z$	$\{\alpha\text{^}i/i\ in\ [1\ldots102]\} \subset GF(2^7)$	102	53	**15**	**15**	**15**	
10	z^3+z^2+z	$\{\alpha^i/i\ in\ [1\ldots103]\} \subset GF(2^7)$	103	82	**7**	**7**	**7**	
11	$z^4+z^3+z^2+z+1$	$\{\alpha^i/i\ in\ [1\ldots103]\} \subset GF(2^7)$	103	75	**9**	**9**	**9**	
12	$z^5+z^4+z^3+z^2+z$	$\{\alpha^i/i\ in\ [1\ldots103]\} \subset GF(2^7)$	103	68	11	11	11	

5. DQSA-DEC DECODER

This section introduces the application of the DQSA method to Goppa code decoding through a hard-decision algorithm named DQSA-Dec. It builds on the analogy between physical principles of DQSA and the optimization process used in decoding. This correspondence is summarized in Table 6 and formalized in Algorithm 3 which describes the implementation of the DQSA-Dec decoder.

Table 6. The analogy between DQSA and DQSA-Dec

DQSA method	DQSA-Dec
PSS state	The information vector's received vector having a **specific** Hamming weight.
ESS state	The information vector's received vector having the **same given** Hamming weight.
Energy (E)	E=value of the hamming weight of a given codeword
Neighbor state	Generating a new information vector having in the case of: 1. **PSS treatment**: a specific Hamming Weight 2. **ESS treatment**: the same Hamming Weight
Temperature	Controlling the decoder via iterations number
final state	final result (Decoded vector ≈ codeword)

Algorithm 3. Which describes the implementation of the DQSA-Dec decoder

```
Inputs:
 1. RV: received vector
 2. tc: error correcting capability of code
 3. Tl_I: Total Iterations by temperature value, Tl_I ϵ [50,2000]
 4. Tᵢ=0.3, T_f=0.002,
 5. δ=0.89
 6. Starting subsystem=PSS
Output: Decoded vector
 1.  Compute the hard decision version of the received vector RV, denoted h
 2.  If the syndrome of h is zero, Then output h as the Decoded vector
      Else
 3.  Determine the information vector U associated with h
 4.  Identify the least reliable positions in RV
 5.  While (Tᵢ > T_f) do:
 6.    For iteration from 1 to Tl_I do:
 7.       If current subsystem is PSS, then generate a neighbor vector U* using PSS
           processing;
 8.       Else generate U* using ESS processing;
 9.       End if
 10.      Compute E(h*), the objective function for the new vector;
 11.      If (E(h*)≤tc+1) or (random(0,1)≤Exp(-E(h*)/Tᵢ)), then update U←U*; h←h*;
 12.      End if
 13.   End For
 14.   With certain probability, switch between PSS and ESS;
 15.   Decrease temperature Tᵢ ← δ*Tᵢ;
 16. End while
 17. Output h as the Decoded vector
End if
```

5.1. Determination of the function (E) to evaluate in algorithm 3 of DSA-Dec

To evaluate the received vector, we determinate the function E as follows:

Let $h = (h_1, \dots, h_n) \epsilon [0,1]^n$ represents the hard-decision version of received vector RV, and $U = (u_1, \dots, u_k)$ $\epsilon\ [0,1]^k$ represents the information vector corresponding to h. For $h^* = (h_1^*, \dots, h_n^*)\ \epsilon\ C$ corresponding to the information vector of h^* is $U^* = (u_1^*, \dots, u_k^*)\ \epsilon\ [0,1]^k$ we define:

$$E(h^*) = \sum_{i=1}^{n} [h_i \oplus (\underset{j=1}{^{k}}\oplus u_j^* g_{ij})] \quad (9)$$

The algorithm of decoder (DSA-Dec) aims to find the information vector U^* corresponding to the codeword h^*, this information vector drives E *(h*)* to a number less or equal to *(tc+1)* of codes.

5.2. Simulation results DQSA-Dec

To validate the efficiency and performance of our DQSA-Dec decoder, we performed a series of numerical simulations on an AWGN channel as shown in Table 7 applied to Goppa codes GC(21,11,5), GC(31,11,9), and GC(62,20,13). These codes, with a coding rate close to 1/2, were tested through multiple trials, varying the number of iterations between 100 and 10,000. The algorithm 2 of decoder, developed in C++, was executed on a Windows 11 computer running on an Intel Core i5 (11th Gen, 2.4 GHz) with 8 GB RAM. All DQSA-Dec performances were compared to the algebraic Patterson decoder [11].

Figure 2 shows that the DQSA-Dec outperforms the Patterson decoder in relation to bit error rate (BER) on all SNR values, offering a 0.6 dB coding gain at BER = 10^{-4}. This highlights the effectiveness of DQSA-Dec, even with only 100 iterations. In Figure 3 we show that the DQSA-Dec decoder outperforms the Patterson decoder in terms of BER across all SNR values, with significant improvements as the number of iterations increases, especially at higher SNR levels. Figure 4 shows that the Patterson decoder outperforms DQSA-Dec for 3000 and 6000 iterations in terms of BER across all SNR levels, but the performance improving for DQSA-Dec with 10,000 iterations achieved the same BER values as Patterson decoder.

Table 7. Simulation parameters for DQSA-Dec

Parameter	Value
Channel Type	AWGN
Modulation Scheme	BPSK
Minimum residual bit errors	200
Minimum transmitted blocks	1500

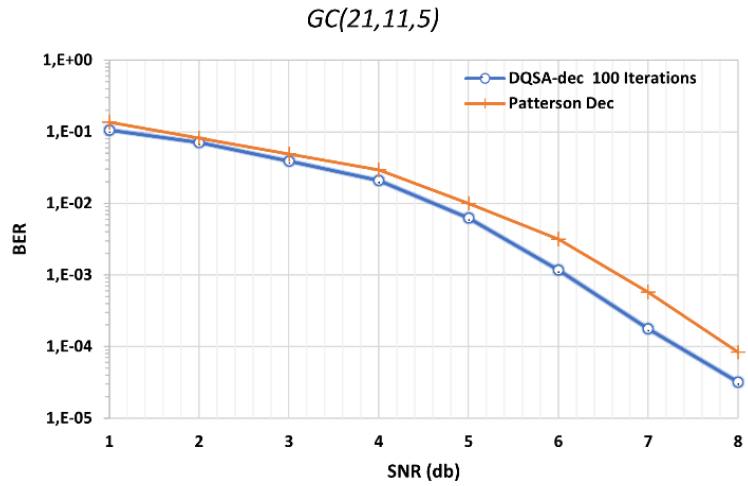

Figure 2. Performance DQSA-Dec Vs Patterson Dec for GC(21, 11, 5)

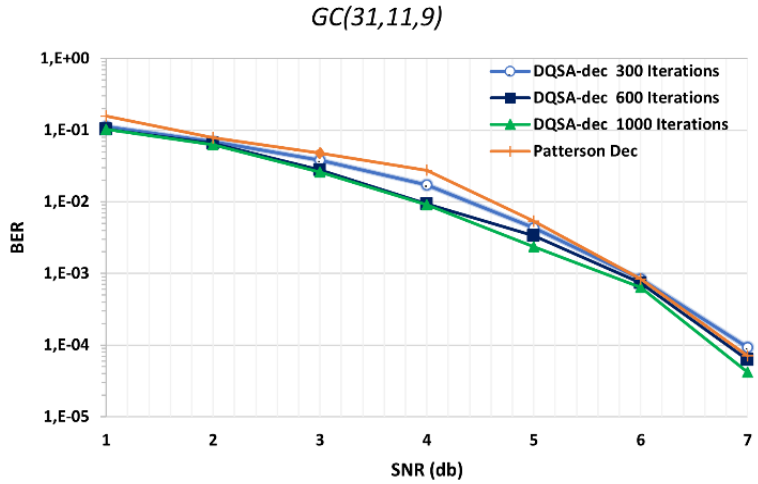

Figure 3. Performance DQSA-Dec Vs Patterson Dec for GC(31, 11, 9)

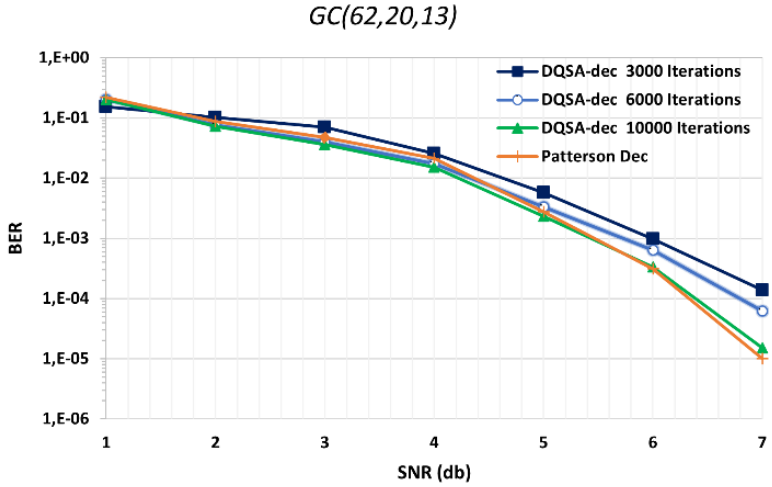

Figure 4. Performance DQSA-Dec Vs Patterson Dec for GC(62, 20, 13)

6. CONCLUSION

This work has demonstrated the effectiveness of the DQSA heuristic method as an alternative and complementary solution, to traditional algebraic techniques, to solve the challenges related to Goppa codes. We proposed two original tools: DQSA-d_{min} for estimating the minimum distance, and DQSA-Dec for hard decoding over AWGN channels. DQSA-d_{min} was validated through close alignment with theoretical distances, and it led to the discovery of 12 optimal Goppa codes that reach the lower bound of d_{min} for linear codes. DQSA-Dec demonstrated superior performance to the Patterson decoder, offering a 0.6 dB coding gain at BER = 10^{-4}, which is notable for a hard-decision decoder. Moreover, both tools exhibited computational efficiency, reducing processing time significantly.

These results pave the way for extending the method to other families of error-correcting codes, further optimizing its parameters, and exploring potential applications in cryptography and quantum error correction systems. Additionally, future work will focus on comparing DQSA to other heuristic approaches and assessing its performance in more complex communication environments.

REFERENCES

[1] W. C. Huffman and V. Pless, *Fundamentals of error-correcting codes*. Cambridge University Press, 2003.
[2] S. Lin and J. Li, *Fundamentals of classical and modern error-correcting codes*. Cambridge University Press, 2021, doi: 10.1017/9781009067928.
[3] T. K. Moon, *Error correction coding: Mathematical methods and algorithms*, 2nd ed. Wiley, 2005, doi: 10.1002/0471739219.
[4] R. H. Morelos-Zaragoza, *The art of error correcting coding: Second edition*, 2nd ed. John Wiley & Sons Ltd., 2006, doi: 10.1002/0470035706.
[5] A. Vardy, "The intractability of computing the minimum distance of a code," *IEEE Transactions on Information Theory*, vol. 43, no. 6, pp. 1757–1766, 1997, doi: 10.1109/18.641542.
[6] K. Knight, "Squibs and discussions: Decoding complexity in word-replacement translation models," *Computational Linguistics*, vol. 25, no. 4, pp. 606–615, Dec. 1999.
[7] E. R. Berlekamp, "Goppa Codes," *IEEE Transactions on Information Theory*, vol. 19, no. 5, pp. 590–592, 1973, doi: 10.1109/TIT.1973.1055088.
[8] V. D. Goppa, *A new class of linear correcting codes*. 1970.
[9] T. Cooklev and A. Yagle, *Modern communications systems*. Michigan Publishing Services, 2024, doi: 10.3998/mpub.14428518.
[10] D. R. Hankerson *et al.*, *Coding theory and cryptography: The essentials, second edition*, 2nd ed. Chapman & Hall/CRC, 2000, doi: 10.1201/b16944.
[11] N. J. Patterson, "The algebraic decoding of Goppa codes," *IEEE Transactions on Information Theory*, vol. 21, no. 2, pp. 203–207, 1975, doi: 10.1109/TIT.1975.1055350.
[12] I. F. Blake and W. C. Huffman, *Algebraic coding theory*. Aegean Park Press, 2013, doi: 10.1201/b15006.
[13] M. Tomlinson, S. V. Bezzateev, M. Jibril, M. A. Ambroze, and M. Z. Ahmed, "Using the structure of subfields in the construction of Goppa codes and extended Goppa codes," *IEEE Transactions on Information Theory*, vol. 61, no. 6, pp. 3214–3224, Jun. 2015, doi: 10.1109/TIT.2015.2419613.
[14] B. Chen and G. Zhang, "The number of extended irreducible binary Goppa codes," *IEEE Transactions on Information Theory*, vol. 69, no. 12. pp. 7691–7711, 2023, doi: 10.1109/TIT.2023.3317290.
[15] X. Li and Q. Yue, "Non-binary irreducible quasi-cyclic parity-check subcodes of Goppa codes and extended Goppa codes," *Designs, Codes, and Cryptography*, vol. 90, no. 7, pp. 1629–1647, 2022, doi: 10.1007/s10623-022-01062-y.
[16] J. Little and H. Schenck, "Codes from surfaces with small Picard number," *SIAM Journal on Applied Algebra and Geometry*, vol. 2, no. 2. pp. 242–258, 2018, doi: 10.1137/17M1128277.
[17] J. L. Carrasquillo-López, A. O. Gómez-Flores, C. Soto, and F. Piñero, "Introducing three best known Goppa codes," *arXiv preprint arXiv:2010.07278*. 2020, doi: 10.48550/arXiv.2010.07278.
[18] I. Byrne, N. Dodson, R. Lynch, E. Pabón–Cancel, and F. Piñero-González, "Improving the minimum distance bound of Trace Goppa codes," *Designs, Codes, and Cryptography*, vol. 91, no. 8, pp. 2649–2663, 2023, doi: 10.1007/s10623-023-01216-6.
[19] S. Bezzateev and N. Shekhunova, "Chain of separable binary goppa codes and their minimal distance," *IEEE Transactions on Information Theory*, vol. 54, no. 12, pp. 5773–5778, 2008, doi: 10.1109/TIT.2008.2006442.
[20] V. Černý, "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm," *Journal of Optimization Theory and Applications*, vol. 45, no. 1, pp. 41–51, 1985, doi: 10.1007/BF00940812.
[21] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983, doi: 10.1126/science.220.4598.671.
[22] B. Aylaj, M. Belkasmi, H. Zouaki, and A. Berkani, "Degeneration simulated annealing algorithm for combinatorial optimization problems," in *International Conference on Intelligent Systems Design and Applications, ISDA*, 2016, vol. 2016-June, pp. 557–562, doi: 10.1109/ISDA.2015.7489177.
[23] B. Aylaj and M. Belkasmi, "Simulated annealing decoding of linear block codes," in *Lecture Notes in Electrical Engineering*, vol. 380, Springer, 2016, pp. 175–183, doi: 10.1007/978-3-319-30301-7_19.
[24] B. Aylaj, M. Belkasmi, S. Nouh, and F. Ayoub, "An innovative method to escape local minima using quantum system degeneracy in simulated annealing," *ARPN Journal of Engineering and Applied Sciences*, vol. 19, no. 24, pp. 1472–1483, 2024, doi: 10.59018/122481.
[25] M. Grassl, "Bounds on the minimum distance of linear codes and quantum codes," *Code Tables*. http://www.codetables.de (accessed May 27, 2024).
[26] A. E. Brouwer, "Bounds on linear codes," in *Handbook of Coding Theory*, V. S. Pless and W. C. Huffman, Eds. Elsevier, 1998, pp. 295–461.
[27] W. Bosma and J. Cannon, *Handbook of Magma functions*, 2.16. 1994.

BIOGRAPHIES OF AUTHORS

Bouchaib Aylaj received his joint Ph.D. degree in computer science from Faculty of Sciences, Chouaib Doukkali University, and ENSIAS, Mohammed V University, Morocco in 2016. He is currently an assistant professor in the Regional Center for Professions of Education and Training (CRMEF)-Rabat Morroco. His research interests include artificial intelligence, error control coding for digital communications, digital signal processing, and embedded systems, He can be contacted at email: bouchaib_aylaj@yahoo.fr.

Said Nouh holds a Ph.D. in computer sciences at National School of Computer Science and Systems Analysis (ENSIAS), Rabat, Morocco in 2014. He is currently professor (higher degree research (HDR)) at Faculty of Sciences Ben M'Sick, Hassan II University, Casablanca, Morocco. His current research interests are artificial intelligence, machine learning, deep learning, telecommunications, information, and coding theory, He can be contacted at email: said.nouh@univh2m.ma.

Mostafa Belkasmi is a professor at ENSIAS (National Higher School of Computer Science and Systems Analysis), Rabat, Morocco; head of information, communication and embedded systems team. He had PhD at Toulouse University in 1991(France). His current research interests include mobile and wireless communications, information and coding theory, He can be contacted at email: mostafa.belkasmi@ensias.um5.ac.ma.