IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 14, No. 4, August 2025, pp. 2776~2787
ISSN: 2252-8938, DOI: 10.11591/ijai.v14.i4.pp2776-2787

Dual simulated annealing soft decoder for linear block codes

Hicham Tahiri Alaoui[1], Ahmed Azouaoui[2], Jamal El Kafi[1]
[1]LaRoSERI Laboratory, Department of Computer Science, Faculty of Sciences, Chouaib Doukkali University, El Jadida, Morocco
[2]Computer Science Research Laboratory (LaRI), Higher School of Technology, Ibn Tofail University, Kenitra, Morocco

Article Info

Article history:

Received Apr 22, 2024
Revised Feb 18, 2025
Accepted Mar 15, 2025

Keywords:

BCH codes
Decoding algorithms
Dual codes
Error correction codes
Linear block codes
Simulated annealing
Soft decoding

ABSTRACT

This paper proposes a new approach to soft decoding for linear block codes called dual simulated annealing soft decoder (DSASD) which utilizes the dual code instead of the original code, using the simulated annealing algorithm as presented in a previously developed work. The DSASD algorithm demonstrates superior decoding performance across a wide range of codes, outperforming classical simulated annealing and several other tested decoders. We conduct a comprehensive evaluation of the proposed algorithm's performance, optimizing its parameters to achieve the best possible results. Additionally, we compare its decoding performance and algorithmic complexity with other decoding algorithms in its category. Our results demonstrate a gain in performance of approximately 2.5 dB at a bit error rate (BER) of 6×10^{-6} for the LDPC (60,30) code.

Corresponding Author:

Hicham Tahiri Alaoui
Department of Computer Science, Faculty of Sciences, Chouaib Doukkali University
El Jadida, Morocco
Email: tahiri-alaoui.h@ucd.ac.ma

1. INTRODUCTION

In digital communication systems, ensuring reliable information transmission over noisy channels is a paramount concern. Error correction techniques [1] play a vital role in addressing this challenge. Among these techniques, linear block codes [2] are particularly noteworthy. These codes add redundant bits to the original message, forming codewords, which enable error detection and correction during transmission. Decoding algorithms are designed to determine the most probable transmitted codeword from the received signal, as shown in Figure 1.

Traditionally, hard-decision decoding algorithms have been employed. However, these algorithms can be computationally intensive and may discard valuable soft information present in the received signal. To address these limitations, researchers have increasingly turned to soft decision decoding algorithms that incorporate the continuous nature of the received signal's amplitude. Soft decision decoding algorithms leverage advanced techniques from information theory, linear algebra, and signal processing to decode received signals with high accuracy. These algorithms are particularly effective in noisy channels, where the reliability of individual bits is uncertain and requires probabilistic treatment. The performance optimization of such algorithms is also a concern to be addressed.

Performance optimization of soft decision decoding algorithms is crucial for achieving improved error correction capabilities. In recent years, metaheuristic and optimization techniques have gained attention for their potential to enhance the performance of these algorithms. Among these techniques, simulated annealing has emerged as a prominent method.

Simulated annealing [3], a technique modeled after the metallurgical annealing process, is well suited for finding near-optimal solutions in complex search spaces. For example, authors in [4], [5] have demonstrated the effectiveness of simulated annealing in improving error correction performance. Additionally, Chen *et al.* [6]

have combined simulated annealing with genetic algorithms, while Niharmine *et al.* [7] introduced a simulated annealing-based algorithm designed for soft decision decoding of linear block codes.

Furthermore, Azouaoui *et al.* [8] developed a new effective technique using the dual code to reduce the complexity of decoding high-rate codes. This approach simplifies the decoding process while maintaining high accuracy. Despite these advancements, the integration of simulated annealing with dual decoding techniques remains underexplored in the literature.

This paper aims to combine the simulated annealing decoder developed by Niharmine *et al.* [7] with the dual decoding technique used by Azouaoui *et al.* [8] to create an efficient and effective error correction method for linear block codes. By integrating these techniques, we seek to enhance decoding performance and improve the data transmission reliability over noisy channels. Surprisingly, this new algorithm gives nearly the same performance as in [7].

The following sections of this paper are organized as follows: section 2 discusses the fundamentals of the simulated annealing algorithm. Section 3 presents our proposed decoder based on the simulated annealing process and duality property. Section 4 examines parameter tuning to find the optimal values to work with, simulation results, and comparisons with main competitors' decoders. Finally, we conclude the paper.

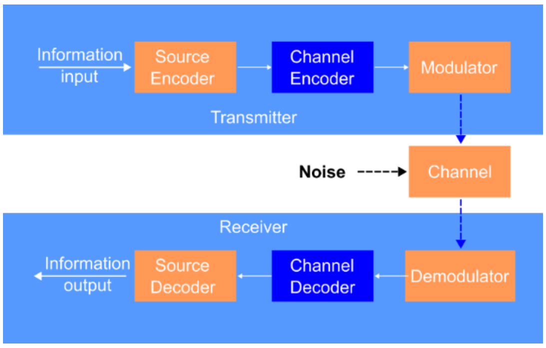


Figure 1. Communication system model

2. COMPREHENSIVE THEORETICAL BASIS

2.1. Basic notations

For the rest of this article, C(n, k, d) will denote a linear code with parameters length n, dimension k, error correction capability t and minimum distance d over the field F_2. This code is representable by a k×n matrix G known as the generator matrix. For the simulated annealing algorithms, N_i is the number of iterations, T_s the starting temperature, T_f the final temperature, α the cooling ratio, S_0 the start solution and N_c the number of iterations required to reach the final temperature T_f. For genetic algorithms, N_i, N_e, and N_g represent the size of the population, the total of elite members, and the generations total, respectively. For the compact genetic algorithm decoder (CGAD) algorithm [9], T_c represents the average number of generations.

2.2. Simulated annealing

Simulated annealing is a versatile metaheuristic algorithm widely used for solving optimization problems. Leveraging the understanding of annealing from the field of metallurgy, it was first introduced by Kirkpatrick *et al.* [3]. The algorithm is particularly effective in finding approximate solutions to both combinatorial and continuous optimization problems. It is also known for its ability to escape local optima, a common issue in optimization algorithms, especially in gradient-based methods. Local optima occur when an algorithm converges to a solution that is optimal within a limited region but not necessarily the global optimum. Simulated annealing addresses this by probabilistically accepting worse solutions, allowing it to explore the solution space more thoroughly. Due to these properties, simulated annealing algorithm has a significant impact in various fields, with applications to combinatorial optimization problems, including but not limited to the traveling salesman problem (TSP) [10]–[12] and the quadratic assignment problem (QAP) [13], [14], very large-scale integration (VLSI) circuit design [15]–[17], to name a few. Here is a description of the simulated annealing algorithm:

a) Initialization: start with an initial solution and set an initial temperature (T) along with a cooling schedule to decrease T over time.

b) Iteration: repeat until a stopping criterion is met (e.g., max iterations or low temperature):

i) Neighborhood generation: apply a perturbation to the current solution, yielding a neighboring solution.

ii) Objective function evaluation: the objective function value of the new solution is calculated to assess its quality. Optimization problems typically aim to minimize this value.
iii) Acceptance or rejection:
– A better new solution is accepted as the current one.
– A worse solution is accepted with a probability exp(-ΔE / T), where ΔE is the difference in objective function values.
c) Cooling: decrease the temperature according to the cooling schedule, which controls the balance between exploration (higher temperature, more acceptance of worse solutions) and exploitation (lower temperature, less acceptance of worse solutions).
d) Termination: stop based on reaching a maximum number of iterations, a specific temperature, or a satisfactory solution quality.
e) Output: return the best solution found.

The subsequent pseudocode illustrates a basic implementation of the simulated annealing algorithm, with all the previous steps:

```
Initialization of parameters (Ni, Ts, Tf, S0)
Set T←Ts   and   S←S0
While (T >Tf)
{
  While (iteration < Ni)
   {
     Select an adjacent solution at random (sn);
     Evaluate ΔEnergy = Energy(sn) – Energy(s);
     If ΔEnergy ≤ 0 then s←sn ;
     else if  random(0,1)  ≤  Exp(-ΔEnergy/T) )  then
        s←sn ; end if;
     end if;
     iteration←iteration+1;
   }
T←cooling(T);
}
```

Key components and parameters of the simulated annealing algorithm include the initial temperature, cooling schedule, neighborhood generation strategy, and acceptance probability calculation. Properly tuning these parameters is crucial to the algorithm's effectiveness in finding high-quality solutions to optimization problems. The approach is explained in more detail in section 3.2.

2.3. Duality property for decoding

To encode a message $m = \{m_i\}_1^k$, we can use (1):

$$c = mG \tag{1}$$

Where c is the codeword. Additionally, in order to determine whether a specific vector is a valid codeword, we introduce a $(n\text{-}k)\times n$ matrix denoted by H (parity-check matrix (PCM)). This particular matrix has the following property:

$$\forall\ v \in F_2^n,\ v \text{ is a codeword if and only if: } Hv^{\mathrm{T}} = 0 \tag{2}$$

Consider the scenario where we transmit a codeword $c=\{c_i\}_1^n$ using BPSK modulation. Let's denote $z=\{z_i\}_1^n$ as the modulated signal transmitted over a Gaussian channel, subject to independent noise components $n=\{n_i\}_1^n$. Here, both z and n are sequences that are statistically independent. Specifically, each n_i is normally distributed with mean 0 and variance N_0 ($n_i \sim \left(\mathrm{N}(0, \frac{N_0}{2})\right)$) where N_0 represents the noise power density.

The received signal, denoted as $r=\{r_i\}_1^n$, is given by the equation $r=z+n$. We introduce $v=\{v_i\}_1^n$ as the binary hard decisions derived from r (quantization of r). Furthermore, we express the error syndrome $s=\{s_i\}_1^{n-k}$ as in (3):

$$s = vH^T \tag{3}$$

In the latter expression the syndrome s is obtained based on the hard decisions made from the received signal r. "$s=0$" means that the received signal corresponds to a valid codeword, indicating an error-free transmission. However, in the presence of transmission errors, our decoder endeavors to determine the codeword ĉ that maximizes the probability P(c|r) within the code space C. Given that all codewords have an equal likelihood of being transmitted, we can write:

$$P(\hat{c}|r) = maxC\ P(c|r) = max\ c\ in\ C\ P(r|c)\ P(c)\ /\ P(r) \quad (4)$$

Considering a discrete memoryless channel with additive white Gaussian noise, where binary antipodal signals are transmitted, with each symbol being independently affected by noise, the maximization of P(r|ĉ) occurs when we minimize the squared norm of the difference between r and ĉ ($\sum_{n=1}^{n}(r_i - c_i)^2$) (or squared Euclidean distance between r and ĉ) as explained in [18], [19]. Consequently, the complex task of maximum-likelihood decoding simplifies to the more straightforward nearest neighbor decoding, using the Euclidean metric. To formalize this reduction, we can express the soft-decision decoding problem as (5) [20]:

$$\begin{aligned}&Given\ a\ received\ word\ r = \{r_i\}_1^n\ ;\\ &find\ a\ codeword\ c\ in\ C\ which\ minimizes\ \sum_{n=1}^{n}(r_i - c_i)^2\end{aligned} \quad (5)$$

This optimization problem involves *n* variables, with only *k* variables as generator base (k most independent and reliable bits).

Instead of solving this optimization problem on the code space, our approach is to search for the error vector, denoted as *e*, that optimizes the solution. To this end, we will construct the error vector *e* through a heuristic method that leverages the dual property [8]. The error *e* has n variables, with only *k* being independent. By using these *k* variables and leveraging the algebraic structure of the code, we can deduce the *k* remaining variables. By doing so, we can deduce, to a certain degree, what was originally sent using (6):

$$c = v + e \quad (6)$$

Since the PCM can be written as H=[A I_{n-k}], where A is a binary matrix of size (n–k)×k, we can write:

$$(v + e)H^T = 0 \Leftrightarrow eH^T = s \quad (7)$$

We define the reliable information set as the collection of the *k* most reliable positions within the received signal $r=\{r_i\}_1^n$. Using this reliability information set, the error vector can be represented as $e = (e_X, e_Y)$, where *X* represents the reliable information set, and Y = {1 n}\X. Consequently, relation (7) can be expressed as (8):

$$(e_X, e_Y)(A^T, I_{n-k}) = s \Leftrightarrow e_Y = e_X A^T + s \quad (8)$$

Having the first part of the error vector (e_X), and using the equation above, we can deduce the second part of the error e_Y and complete the whole error vector $e = (e_X, e_Y)$. We can then verify that $(v+e)$ is a valid codeword.

Our endeavor is to search, amongst all the error set Err(s) with syndrome *s*, for the error vector *e* which minimizes the squared norm of the difference between the received signal *r* and the related codeword c=(v+e) which is given by (9):

The squared norm of the difference between r and c: $\sum_{n=1}^{n}(r_i - c_i)^2$ (9)

Taking into account the analysis provided earlier, the algorithm's description and steps are outlined in the following section.

3. THE DUAL SIMULATED ANNEALING SOFT DECODER ALGORITHM

3.1. Construction of the DSASD algorithm

Building on the previous section, where we outlined the two fundamental components of our proposed algorithm, the next section presents the dual simulated annealing soft decoder (DSASD). Table 1 outlines the complete mapping between the proposed algorithm and the physical simulated annealing. The energy is represented as the Euclidean distance between the received word and a codeword, and the state is represented as a k-bit vector. The lowest energy state corresponds to the nearest codeword.

Table 1. Mapping between the proposed algorithm (DSASD) and physical simulated annealing algorithm

Physical simulated annealing	DSASD
Energy	The Euclidean distance between a codeword and the received word
State	k-bit vector
Final state	The decoded word

The DSASD algorithm is described as follows:

i) Step 1: randomly generate and encode k binary information bits using the code's matrix G, resulting in an n-bit vector. After transforming each *0* to a *1* and each *1* to a *-1*, introduce simulated Gaussian noise to produce a received vector, denoted as r, where r belongs to $\mathbb{R}^n$.
ii) Step 2: after receiving the sequence $r=\{r_i\}_1^n$, make a binary hard decision for this received signal $r=\{r_i\}_1^n$ to obtain $v=\{v_i\}_1^n$:

$$v_i = \begin{cases} 1, & r_i < 0 \\ 0, & r_i \geq 0 \end{cases}$$

iii) Step 3: compute the syndrome as $s=vH^T$. If s equals zero, output v and terminate; otherwise, proceed further.
iv) Step 4: apply a permutation to the coordinates of the received vector r, ensuring that the last $(n-k)$ positions hold the least reliable linearly independent components of r.
   – r_i is considered more reliable than r_j if $|r_i| > |r_j|$ based on the assumption that the data is corrupted by additive white Gaussian noise during transmission.
   – Sort the sequences $r=\{r_i\}_1^n$ in descending order of reliability, then apply a second permutation that lets the last n-k elements of r be the least linearly independent elements, to create new sequences $r'=\{r'_i\}_1^n$. Let's denote π as the permutation mapping $r'=\pi(r)$.
   – Apply this permutation π to H to obtain H' ($H'=\pi(H)$) and $v'(v'=\pi(v))$.
   – Use Gaussian elimination on H' to derive a systematic matrix.
v) Step 5: Generate an error vector of k-bits with s as syndrome:
   – The first generated error vector can be the zero vector.
   – Randomly generate an error vector e_X of k-bits.
   – The second part of the vector is $e_Y=e_XA^T+s$
   – The error vector e is formed as (e_X, e_Y)
vi) Step 6: apply the simulated annealing algorithm and use (8) to get the best error candidate e_{best}. This algorithm is depicted afterwards.
vii) Step 7: Obtain the codeword:
   – The obtained codeword $c'=v'+e_{best}$ is associated with the matrix H', hence we estimate the codeword $\hat{c}$ to be:

$$\hat{c} = \pi^{-1}(c') \tag{10}$$

A simulated annealing algorithm, using squared Euclidean distance as its objective function, finds the closest codeword. Our simulated annealing method, used in step 6, differs from classical simulated annealing by using reliability information to guide solution generation, rather than random bit flipping as follows:

```
select_neighbor() {
       For each bit i = 1 to k
       If (Random (between 0 and 1) > 1/(1+exp(-2 r'_i/N_0)) ) then (switch the
bit r'_i)
       End for    }
```

Finally, the DSASD algorithm can be represented with the following pseudocode:

```
Set the parameters {Ni ,Ts ,Tf, α, S0}; Set   T  = T0   and   S  = S0
While (T >Tf) {
       While (iteration < Ni) {
              Sn      = select_neighbor();
              ΔEnergy = Energy(Sn) - Energy(S);
              error   = EvaluateCorrectedError();
              if (error < T) then break;
              if ΔEnergy ≤ 0 then S = Sn;
              else if random(0,1) ≤ Exp(-ΔEnergy/T) then
       S = Sn ; end if
              end if
              iteration      = iteration + 1;
       }
       T = α * T
}
```

where energy(s)=$\sum_{n=1}^{n}(r'_i - c_i)^2$, and c={$c_i$}$_1^n$ is the related codeword.

3.2. DSASD algorithm parameter tuning

Optimizing our algorithm's parameters {α_i, N_i, T_s, T_f} is a key challenge. While probabilistic models like MacKay's [21] could be used, we instead conduct multiple simulations due to our assumption of parameter independence. We evaluate bit error rate (BER) against signal to noise ratio (SNR), varying one parameter at a time while holding the others at their default values, as outlined in Table 2. The variations of parameters α and Ni are illustrated in Figure 2, which consists of two sub-figures:

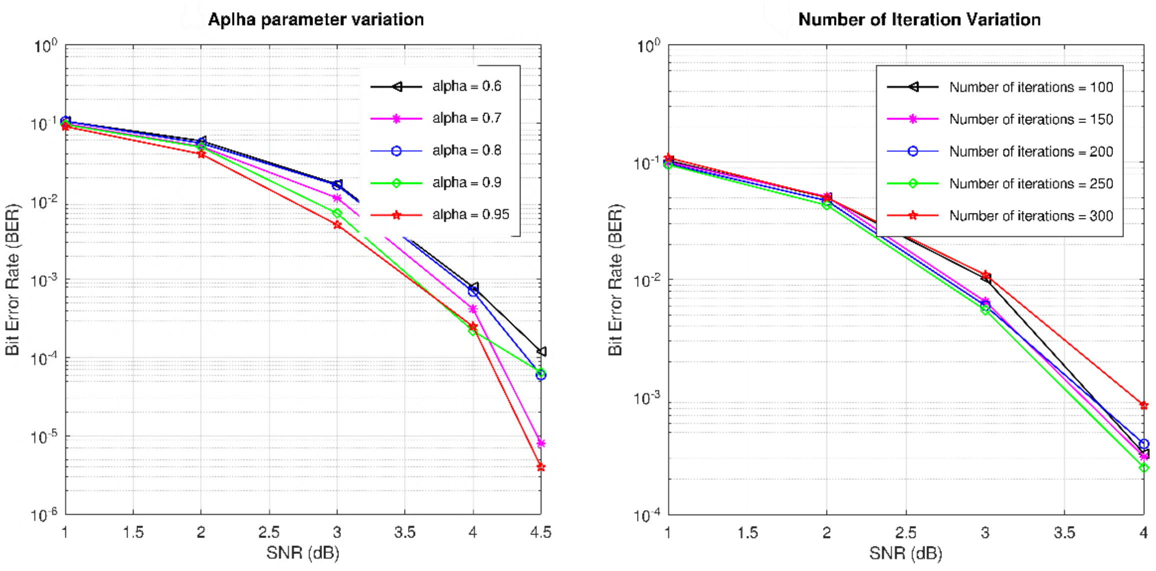
- Parameter α: analyzing Figure 2(a) shows that choosing α=0.95 is an effective option for the cooling ratio, approaching the best performance. In practical applications, a deliberate slow cooling mechanism is beneficial, as it helps in identifying and utilizing codewords with low Euclidean distance.
- Parameter N_i: we typically determine the optimal number of iterations experimentally. Our simulations, shown in Figure 2(b), indicate that setting N_i to 250 yields near-optimal results.

The effects of parameters Ts and Tf on system performance are depicted in Figure 3, comprising two sub-figures:

- Parameter T_s: examining the simulation results presented in Figure 3(a), it is evident that the optimal initial temperature T_s is 0.2. Given the significant impact of this parameter on accuracy, various methods for estimating T_s effectively have been proposed, such as those introduced in [22].
- Parameter T_f: in the context of physics, the concept of a freezing temperature is intuitively used to achieve equilibrium. Our simulations, as shown in Figure 3(b), confirm this idea, demonstrating that optimal performance is consistently attained across various SNR values when the temperature decreases to T_f=0.001.

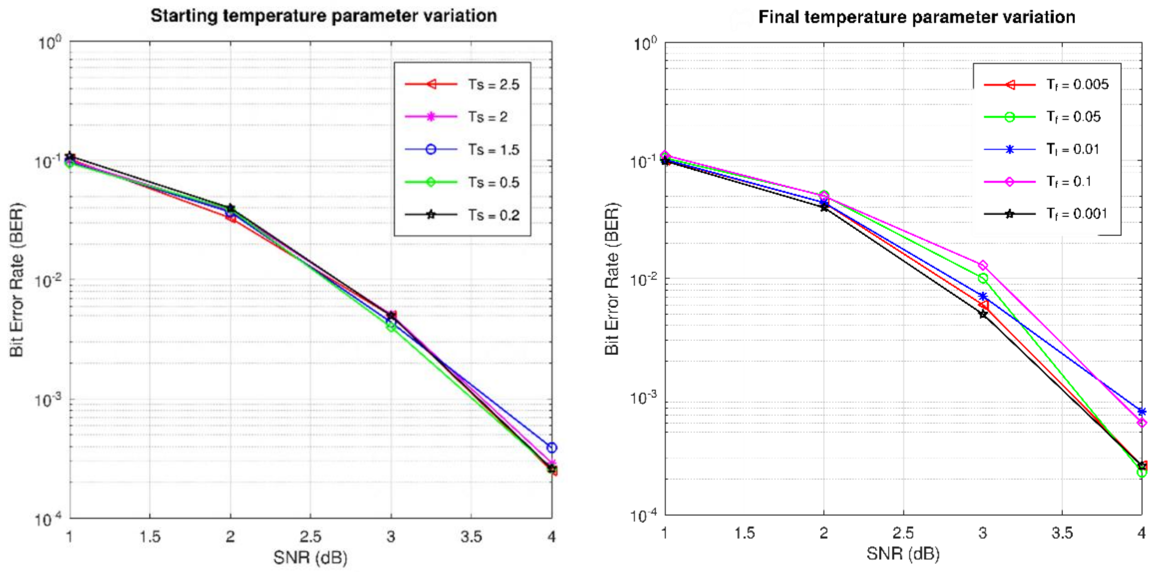
Table 2. Parameter values of the DSASD algorithm

Parameter	Value
Default code	BCH(63,45,7)
Channel	AWGN
Modulation	BPSK
Minimum number of bit errors	200
Minimum number of blocks	1000
N_i	250
T_s	0.2
T_f	0.001
α	0.95



(a) (b)

Figure 2. Variation of (a) parameter α and (b) number of iterations



(a) (b)

Figure 3. Evolution of (a) parameter T_s and (b) parameter T_f

3.3. Complexity analysis

DSASD algorithm's step 4 has $O(k^2n)$ time complexity [23]. However, parallelization can reduce the time complexity to O(kn), a cost negligible compared to step 6's (N_iN_c nk) time complexity. As shown in Table 3, the computational cost of the Chase-2 [24] and soft decoding based genetic algorithm (SDGA) [25] algorithms scales exponentially with t. Therefore, codes with high error correction capabilities present the most challenging computational complexity and tend to exhibit suboptimal performance as n increases. Conversely, the DSASD, simulated annealing soft decoder (SASD) [7], dual domain decoding genetic algorithm (DDGA) [8], CGAD [9], Maini [20], and genetic algorithm for decoding systematic block codes (AutDAG) [26] algorithms exhibit linear complexity, with respect to either n or k.

Table 3. Comparison of the DSASD algorithm complexity with others

Parameter	Value
Chase-2	$O(2^t\, n^2\log_2(n))$
DDGA	$O(N_iN_g\,[k(n-k)+\log(N_i)])$
Maini	$O(N_iN_g\,[kn+\log(N_i)])$
AutDAG	$O(N_iN_g\,kn)$
SDGA	$O(2^t\,(N_iN_g\,[kn^2+kn+\log(N_i)]))$
CGAD	$O(T_c\,k(n\text{-}k))$
SASD	$O(N_iN_c\,kn)$
DSASD	$O(N_iN_c\,nk)$

4. RESULTS AND DISCUSSION

This study examines the impact of combining the dual property with simulated annealing in the DSASD algorithm. Although previous research has applied the dual property with various decoders, this specific combination has not been previously explored. The DSASD algorithm was implemented in C, with figures generated using octave [27]. Our simulations were performed on a workstation with an Intel Core(TM) i7-6920HQ processor with 16 GB of memory clocked at 2.90 GHz, running Ubuntu 18.04.6 LTS x86_64 operating system. Optimal values for algorithm parameters were selected as outlined in section 3.2. Performance was assessed based on BER as a function of SNR (Eb/N0).

We conducted a comparative analysis of the proposed DSASD algorithm against classical simulated annealing, SASD [7], and other decoders in the same category. The simulations were performed using the default parameters specified in Table 2. Starting from the next paragraph, we detail the performance of

DSASD compared to a set of other decoders, including SASD [7], classical simulated annealing, Chase-2 [24], DDGA [8], CGAD [9], Maini [20], compact genetic algorithm with high selection pressure (cGA-HSP) [28], AutDAG [26], compact genetic algorithm with memory (cGA-M) [29], genetic algorithm meta-decision decoder (GAMD) [30], SDGA [25], and classical binary phase shift keying (BPSK) decoding algorithms.

The comparison of DSASD, SASD, and classical simulated annealing for BCH(31,21,5), BCH(63,45,7), and LDPC(60,30) codes is shown in Figure 4. This comparison underscores the superior efficacy of both the DSASD and SASD algorithms over the classical method for these codes. Specifically, Figure 4(a) illustrates the results for BCH(31,21,5), Figure 4(b) for BCH(63,45,7), and Figure 4(c) for LDPC(60,30).

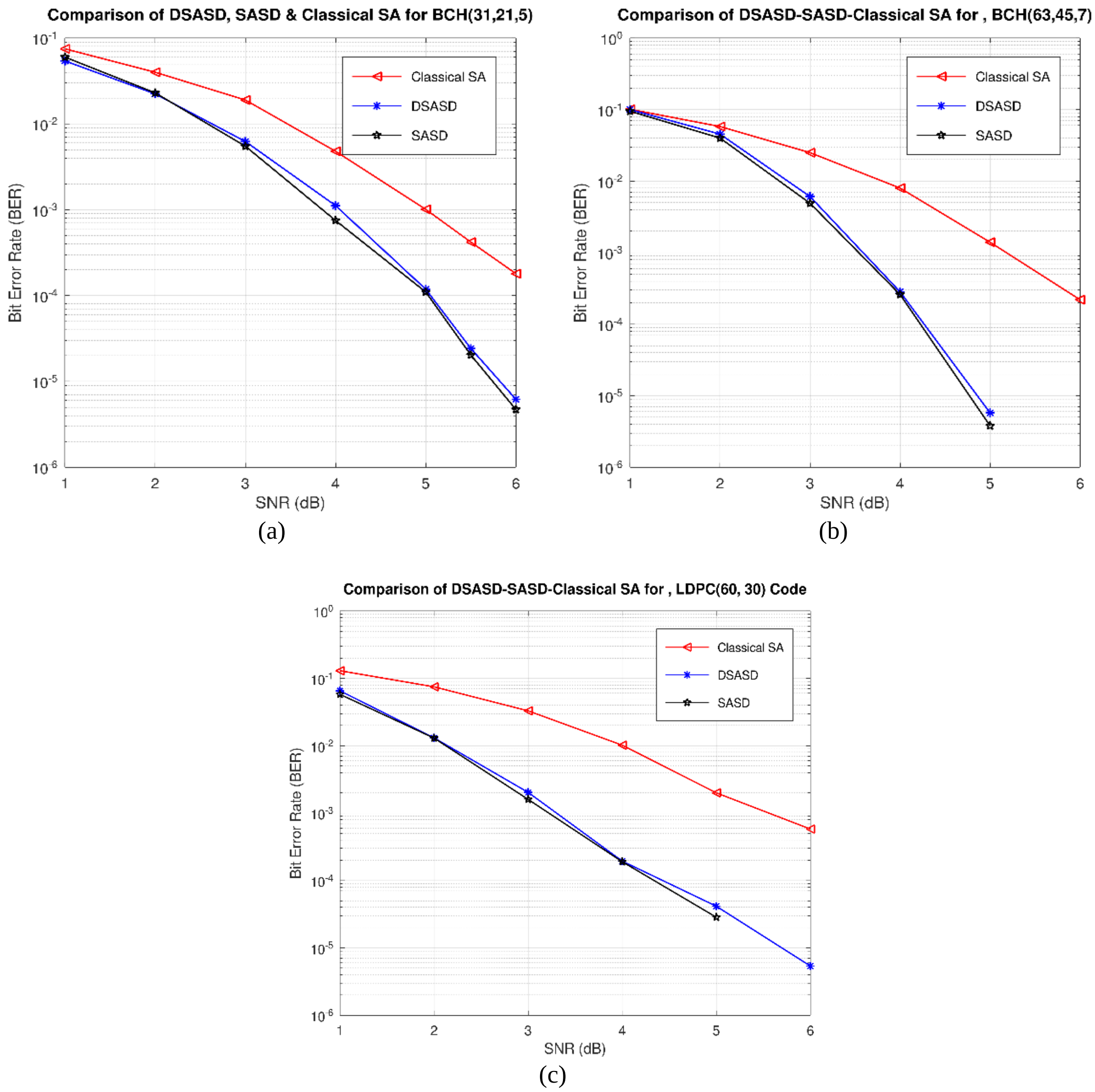
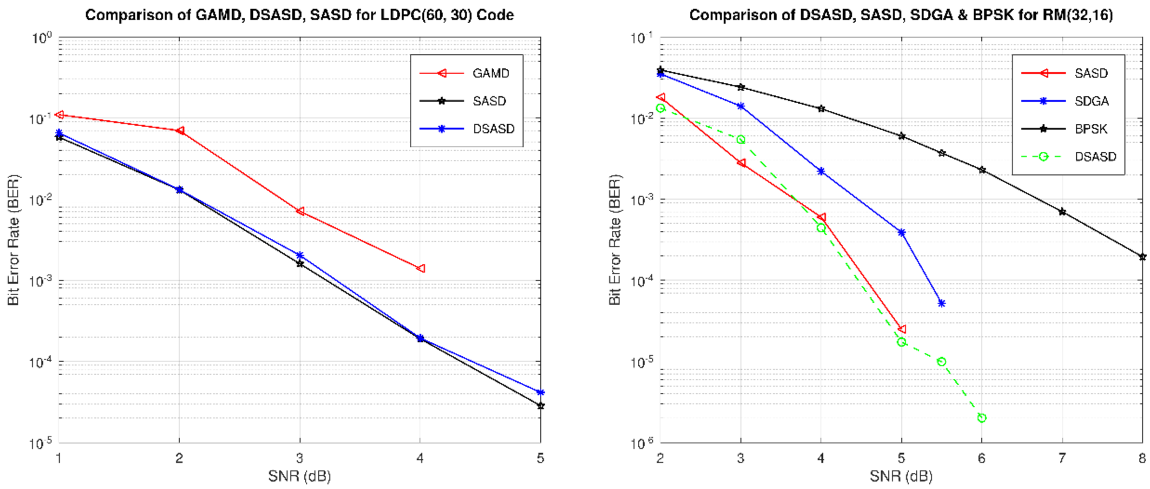


Figure 4. Comparison of DSASD, SASD, and classical simulated annealing for the (a) BCH(31,21,5), (b) BCH(63,45,7), and (c) LDPC(60,30)

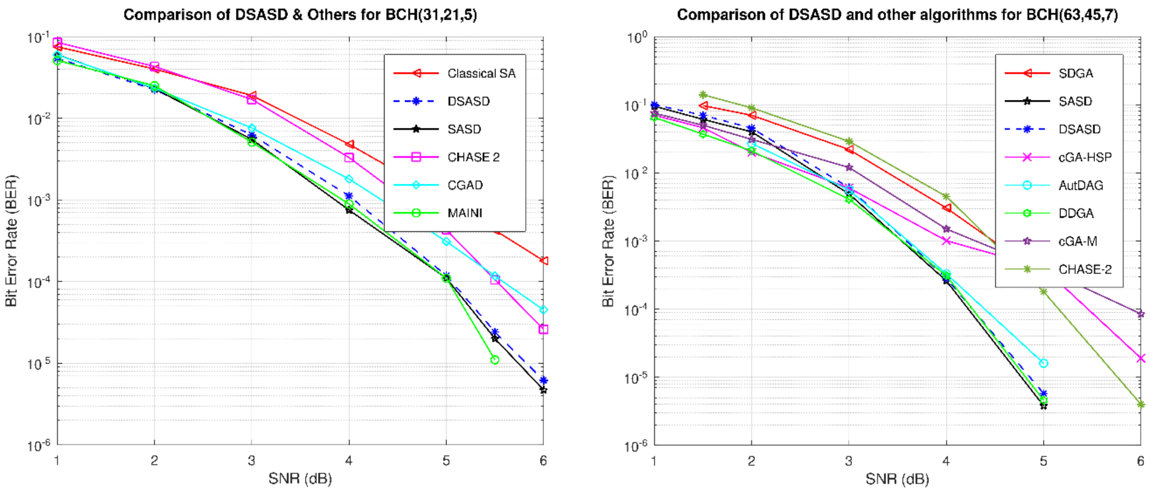
The evaluation of DSASD decoding performance against various competitor algorithms is presented in Figure 5. Figure 5(a) demonstrates that the DSASD algorithm outperforms other algorithms by 1 dB at 10^{-3} over the GAMD [30] algorithm for the LDPC(60,30) code, and Figure 5(b) shows that DSASD achieves a 0.25 dB gain over SDGA at 10^{-4} and about a 1.83 dB gain at 10^{-3} over BPSK decoding for the RM(32,16,8) code. The performance comparison of DSASD and other competing algorithms on BCH codes is presented in

Figure 6. For BCH(31,21,5), as shown in Figure 6(a), the DSASD algorithm surpasses classical simulated annealing (by 1 dB starting from 10^{-3}), Chase-2 and CGAD by 0.5 dB at 10^{-4}, and produces results nearly identical to those of SASD and Maini, except for Maini with a 0.31 dB difference at 10^{-5}. Similar superior performance is observed for the BCH(63,45,7) code in Figure 6(b), where DSASD outperforms SDGA, Chase-2, cGA-M, cGA-HSP, and AutDAG, with comparable performance to DDGA [8] and SASD [7].



(a) (b)

Figure 5. Evaluation of DSASD decoding performance against competitor algorithms applied to: (a) LDPC(60,30) and (b) RM(32,16,8)



(a) (b)

Figure 6. Performance comparison of DSASD and competing algorithms on BCH codes: (a) BCH(31,21,5) and (b) BCH(63,45,7).

The simulation results clearly indicate that the proposed DSASD algorithm offers superior performance compared to classical simulated annealing and many other decoders tested across various codes. Notably, the DSASD achieves significant gains, such as up to 2 dB for BCH(63,45,7) and LDPC(60,30), demonstrating consistent improvement over other methods. Additionally, the close alignment in outcomes between SASD and DSASD further validates the robustness and effectiveness of the proposed algorithm. The superior performance of the DSASD algorithm can be attributed to the effective integration of the dual property with simulated annealing, which can be noticed on large codes. The consistency of results across different codes indicates that the proposed method is robust and adaptable to a range of error correction scenarios.

This study demonstrated the successful integration of simulated annealing with the dual property of linear block codes. While promising, further research is required to confirm its applicability to other codes and to improve convergence. Specifically, optimizing the cooling schedule could address slow convergence issues, and adapting the decoder for broader code compatibility could enhance the algorithm’s overall effectiveness and versatility. These efforts will be key to overcoming the current limitations and expanding the DSASD algorithm's potential.

Finally, this study highlights the DSASD algorithm's superior performance, achieving up to 2 dB improvement over classical simulated annealing and other decoders, particularly for large codes like BCH(63,45,7) and LDPC(60,30). The successful integration of simulated annealing with the dual property underscores its robustness. However, further research is needed to improve convergence and extend its applicability to a wider range of codes, ensuring the DSASD algorithm reaches its full potential.

5. CONCLUSION

In this paper, we introduced a new soft decoder for linear block codes by integrating the simulated annealing process with the duality property of linear block codes. The simulated annealing algorithm uses a probabilistic approach to explore the solution space, drawing inspiration from the annealing process. Suboptimal solutions are accepted according to a temperature schedule. Our proposed DSASD surpasses classical simulated annealing and other decoders such as SDGA, AutDAG, Chase-2, and CGAD for specific codes, achieving gains of up to 2.5 dB at a BER of 6×10-6 for the LDPC(60, 30) code. A key advantage of DSASD is its ability to leverage reliability information from received data to initiate the search and generate neighboring solutions. Additionally, we conducted a comparative analysis of algorithmic complexity, highlighting DSASD's efficiency. Future work should focus on enhancing the convergence of the cooling mechanism and extending the algorithm’s applicability to a broader range of codes.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Hicham Tahiri Alaoui	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓		✓	
Ahmed Azouaoui	✓	✓	✓	✓		✓	✓		✓	✓	✓	✓	✓	
Jamal El Kafi		✓		✓			✓	✓		✓	✓	✓	✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

INFORMED CONSENT

Not applicable. No personal information was included in this study.

ETHICAL APPROVAL

Not applicable. No research related to human use has been done in this paper.

DATA AVAILABILITY

Data availability is not applicable to this study, as the data analyzed consist solely of randomly generated binary message sequences used for codeword simulation. These can be reproduced by any researcher using the described algorithm.

REFERENCES

[1] D. J. Costello, J. Hagenauer, H. Imai, and S. B. Wicker, "Applications of error-control coding," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2531–2560, 1998, doi: 10.1109/18.720548.

[2] B. Sklar and F. J. Harris, "The ABCs of linear block codes," *IEEE Signal Processing Magazine*, vol. 21, no. 4, pp. 14–35, 2004, doi: 10.1109/MSP.2004.1311137.

[3] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983, doi: 10.1126/science.220.4598.671.

[4] A. E. Gamal, L. Hemachandra, I. Shperling, and V. Wei, "Using simulated annealing to design good codes," *IEEE Transactions on Information Theory*, vol. 33, no. 1, pp. 116–123, 1987, doi: 10.1109/TIT.1987.1057277.

[5] B. Aylaj and M. Belkasmi, "Simulated annealing decoding of linear block codes," in *Proceedings of the Mediterranean Conference on Information & Communication Technologies 2015*, Apr. 2016, vol. 380, pp. 175–183, doi: 10.1007/978-3-319-30301-7_19.

[6] H. Chen, N. S. Flann, and D. W. Watson, "Parallel genetic simulated annealing: a massively parallel SIMD algorithm," *IEEE Transactions on Parallel and Distributed Systems*, vol. 9, no. 2, pp. 126–136, 1998, doi: 10.1109/71.663870.

[7] L. Niharmine, H. Bouzkraoui, A. Azouaoui, and Y. Hadi, "Simulated annealing decoder for linear block codes," *Journal of Computer Science*, vol. 14, no. 8, pp. 1174–1189, 2018, doi: 10.3844/jcssp.2018.1174.1189.

[8] A. Azouaoui, M. Belkasmi, and A. Farchane, "Efficient dual domain decoding of linear block codes using genetic algorithms," *Journal of Electrical and Computer Engineering*, vol. 2012, no. 1, 2012, doi: 10.1155/2012/503834.

[9] A. Azouaoui, A. Berkani, and P. M. Belkasmi, "An efficient soft decoder of block codes based on compact genetic algorithm," *arXiv-Computer Science*, pp. 1–8, 2012.

[10] X. Geng, Z. Chen, W. Yang, D. Shi, and K. Zhao, "Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search," *Applied Soft Computing Journal*, vol. 11, no. 4, pp. 3680–3689, 2011, doi: 10.1016/j.asoc.2011.01.039.

[11] E. H. L. Aarts, J. H. M. Korst, and P. J. M. van Laarhoven, "A quantitative analysis of the simulated annealing algorithm: A case study for the traveling salesman problem," *Journal of Statistical Physics*, vol. 50, no. 1–2, pp. 187–206, 1988, doi: 10.1007/BF01022991.

[12] S. Zhan, J. Lin, Z. Zhang, and Y. Zhong, "List-based simulated annealing algorithm for traveling salesman problem," *Computational Intelligence and Neuroscience*, vol. 2016, pp. 1–12, 2016, doi: 10.1155/2016/1712630.

[13] A. Misevičius, "A modified simulated annealing algorithm for the quadratic assignment problem," *Informatica*, vol. 14, no. 4, pp. 497–514, 2003, doi: 10.15388/Informatica.2003.037.

[14] M. R. Wilhelm and T. L. Ward, "Solving quadratic assignment problems by 'simulated annealing,'" *IIE Transactions*, vol. 19, no. 1, pp. 107–119, 1987, doi: 10.1080/07408178708975376.

[15] J. A. Chandy and P. Banerjee, "Parallel simulated annealing strategies for VLSI cell placement," in *Proceedings of 9th International Conference on VLSI Design*, 1996, pp. 37–42, doi: 10.1109/ICVD.1996.489451.

[16] J. Chen, W. Zhu, and M. M. Ali, "A hybrid simulated annealing algorithm for nonslicing VLSI floorplanning," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 41, no. 4, pp. 544–553, 2011, doi: 10.1109/TSMCC.2010.2066560.

[17] D. Kolar, J. D. Puksec, and I. Branica, "VLSI circuit partition using simulated annealing algorithm," in *Proceedings of the 12th IEEE Mediterranean Electrotechnical Conference (IEEE Cat. No.04CH37521)*, 2004, pp. 205–208, doi: 10.1109/MELCON.2004.1346809.

[18] K. Farrell, L. Rudolph, C. Hartmann, and L. Nielsen, "Decoding by local optimization (Corresp.)," *IEEE Transactions on Information Theory*, vol. 29, no. 5, pp. 740–743, 1983, doi: 10.1109/TIT.1983.1056724.

[19] G. C. Clark and J. B. Cain, *Error-correction coding for digital communications*. Boston, MA: Springer United States, 1981, doi: 10.1007/978-1-4899-2174-1.

[20] H. Maini, K. Mehrotra, C. Mohan, and S. Ranka, "Genetic algorithms for soft-decision decoding of linear block codes," *Evolutionary Computation*, vol. 2, no. 2, pp. 145–164, 1994, doi: 10.1162/evco.1994.2.2.145.

[21] D. J. C. MacKay, *Information theory, inference and learning algorithms*. Cambridge, United Kingdom: Cambridge University Press, 2003, doi: 10.1017/S026357470426043X.

[22] W. Ben-Ameur, "Computing the initial temperature of simulated annealing," *Computational Optimization and Applications*, vol. 29, no. 3, pp. 369–385, 2004, doi: 10.1023/B:COAP.0000044187.23143.bd.

[23] Y. S. Han, C. R. P. Hartmann, and C.-C. Chen, "Efficient maximum-likelihood soft-decision decoding of linear block codes using algorithm A*," in *IEEE International Symposium on Information Theory*, San Antonio, United States, 1993, pp. 27–27, doi: 10.1109/ISIT.1993.748342.

[24] D. Chase, "Class of algorithms for decoding block codes with channel measurement information," *IEEE Transactions on Information Theory*, vol. 18, no. 1, pp. 170–182, 1972, doi: 10.1109/TIT.1972.1054746.

[25] A. Azouaoui and M. Belkasmi, "A new genetic decoding of linear block codes," in *2012 International Conference on Multimedia Computing and Systems*, 2012, pp. 1176–1182, doi: 10.1109/ICMCS.2012.6320254.

[26] S. Nouh, I. Chana, and M. Belkasmi, "Decoding of block codes by using genetic algorithms and permutations set," *International Journal of Communication Networks and Information Security*, vol. 5, no. 3, pp. 201–209, Dec. 2013, doi: 10.17762/ijcnis.v5i3.428.

[27] J. W. Eaton, D. Bateman, S. Hauberg, and R. Wehbrin, "GNU Octave version 5.2.0 manual: a high-level interactive language for numerical computations," *Octave*. 2020. [Online]. Available: https://www.octave.org/

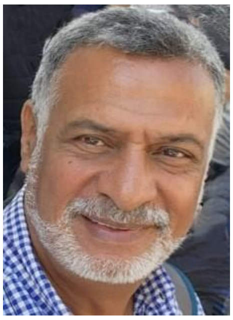
[28] A. Berkani, A. Azouaoui, and M. Belkasmi, "Soft-decision decoding by a compact genetic algorithm using higher selection pressure," in *2015 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, 2015, pp. 1–6, doi: 10.1109/WINCOM.2015.7381308.

[29] A. Berkani, A. Azouaoui, M. Belkasmi, and B. Aylaj, "Improved decoding of linear block codes using compact genetic algorithms with larger tournament size," *International Journal of Computer Science Issues*, vol. 14, no. 1, pp. 15–24, 2017, doi: 10.20943/01201701.1524.
[30] A. G. Scandurra, A. L. Dai Pra, L. Arnone, L. Passoni, and J. C. Moreira, "A genetic-algorithm based decoder for low density parity check codes," *Latin American Applied Research*, vol. 36, no. 3, pp. 169–172, 2006.

BIOGRAPHIES OF AUTHORS

Hicham Tahiri Alaoui holds an engineering degree from Ecole Nationale Supérieure d'Informatique et d'Analyse des Systèmes (ENSIAS), Morocco in 2006. He is currently pursuing a Ph.D. degree at the Faculty of Science, Chouaib Doukkali University, Morocco. His research includes information theory, information security, cryptanalysis, deep learning, and machine learning. He can be contacted at email: hicham.ta@gmail.com or tahiri-alaoui.h@ucd.ac.ma.

Ahmed Azouaoui received his license in computer science and engineering in June-2001 and master in computer science and telecommunication from University of Mohammed V - Agdal, Rabat, Morocco in 2003. He received his Ph.D. in computer science in 2014 and engineering at Department of Computer Science, National School of Computer Science and Systems Analysis (ENSIAS), Rabat, Morocco. Currently, he is a professor at Higher School of Technology, Ibn Tofail University, Morocco. His areas of interest are information systems, coding theory, and artificial intelligence. He can be contacted at email: a.azouaoui@uit.ac.ma.



Jamal El Kafi seasoned academic with a Ph.D. in robotics and certifications in quality auditing and coaching. As a full professor at Chouaib Doukkali University, he has established himself as a leading expert in artificial intelligence, ICT, and educational technologies. He heads the DIS research team, focusing on decision support systems, and is an associate member of the ELITES laboratory. He is the founder and president of TENORS, an association dedicated to promoting new technologies and scientific research. With extensive experience in AI, IS, quality management, and international education systems, he has authored numerous scientific articles, evaluated research papers, and supervised multiple Ph.D. theses. He can be contacted at email: elkafi.j@ucd.ac.ma or jelkafi@gmail.com.