Iterative Soft-Decision Decoding of Hermitian Codes

Li Chen
School of Information Science and Technology, Sun Yat-sen University,
Guangzhou, China. Email: chenli55@mail.sysu.edu.cn

***Abstract*—Algebraic-geometric (AG) codes have long been identified as a possible candidate to replace Reed-Solomon (RS) codes for error-correction. This paper proposes an iterative soft-decision decoding algorithm for one of the most popular AG codes – Hermitian codes. The algorithm is designed by integrating the legacy belief propagation (BP) algorithm and the Koetter-Vardy (KV) soft-decision list decoding algorithm. The BP algorithm performs iterative decoding based on an adapted parity-check matrix whose density has been reduced, namely the adaptive BP (ABP) algorithm. It enhances the reliability of the received information, with which the KV algorithm performs soft-decision list decoding to obtain the intended message. Since the matrix adaptation is bit reliability oriented, re-grouping of the unreliable bits is introduced to assist the ABP algorithm. Geometric analysis of the ABP algorithm is presented, demonstrating the necessity of performing matrix adaptation and integrating the ABP and KV algorithms. The performance evaluation shows the proposed iterative decoding algorithm is an advanced decoding approach that outperforms the existing decoding algorithms for Hermitian codes. It can also outperform ABP-KV decoding of RS codes.**

***Index Terms*—Adaptive belief propagation, algebraic-geometric codes, Hermitian codes, iterative decoding, Koetter-Vardy algorithm.**

I. Introduction

Hermitian codes are the most celebrated algebraic-geometric (AG) codes, which are believed to have the potential to replace Reed-Solomon (RS) codes for error-correction. Compared with the RS codes that are defined over the same finite field, the Hermitian codes are longer and hence inherit a better error-correction capability [1] [2].

The pioneer work on efficient decoding of Hermitian codes was proposed by Sakata *et al.* [3]. Combined with the majority voting algorithm [4], the Sakata algorithm can correct symbol errors up to half of the code's designed minimum distance. Guruswami and Sudan [5] proposed a hard-decision list decoding algorithm (or the so-called GS algorithm) for both RS and AG codes, correcting symbol errors beyond the half distance bound. Hard-decision list decoding of Hermitian codes was proposed by Hoholdt *et al.* [6], followed by an efficiency improved decoding proposed by Chen *et al.* [7]. Recently, soft-decision list decoding of Hermitian codes was introduced by Chen *et al.* [2] and Lee *et al.* [8] independently.

In the meantime, soft-decision decoding of RS codes is an active area of research. Related works include the maximal likelihood (ML) decoding [9], the ordered statistics decoding [10] and the Koetter-Vardy (KV) list decoding [11]. Recently, iterative soft-decision decoding of RS codes using the adaptive belief propagation (ABP) algorithm was proposed by Jiang *et al.* [12]. By incorporating the ABP and KV algorithms, El-Khamy *et al.* [13] proposed an improved iterative soft-decision decoding algorithm for RS codes. It is shown that the ML decoding performance bound is approached with a moderate decoding complexity.

However, iterative soft-decision decoding of Hermitian codes is yet to be developed. So far, their best error-correction performance was achieved by the KV algorithm [2] [8]. To explore the error-correction potential of Hermitian codes, more advanced decoding approach would be desirable. This paper proposes the first iterative soft-decision decoding algorithm for Hermitian codes, namely the ABP-KV algorithm. The ABP algorithm performs the first stage decoding to enhance the reliability of the received information. They will then be passed to the second stage decoding, the KV algorithm. The parity-check matrix of Hermitian codes is defined and its Gaussian elimination will be deployed based on the knowledge of bit reliabilities. It reduces the density of the matrix and eliminates part of its short cycles. To improve the performance, re-grouping of the unreliable bits is also introduced. Geometric analysis of the ABP algorithm is presented, demonstrating the necessity of performing the matrix adaptation and integrating the ABP and KV algorithms. Performance evaluation of the proposed algorithm will also be provided, comparing it with the existing decoding approaches for Hermitian codes and the ABP-KV decoding of RS codes.

II. Construction of Hermitian Codes

Let $\mathbb{F}_q$ denote the finite field of size q and $\mathbb{F}_q = \{0, 1, \alpha, \alpha^2, \ldots, \alpha^{q-2}\}$, where α is a primitive element. In this paper, it is assumed that q is a square such that $q = 2^{\varpi}$ where ϖ is an even number. Let $\mathbb{F}_q[x]$, $\mathbb{F}_q[x, y]$ and $\mathbb{F}_q[x, y, z]$ denote the rings of univariate, bivariate and trivariate polynomials defined over $\mathbb{F}_q$, respectively. The Hermitian curve that is defined in $\mathbb{F}_q$ can be written as [7]:

$$H_w(x, y, z) = x^{w+1} + y^w z + y z^w, \tag{1}$$

where $w = \sqrt{q}$. The construction of a Hermitian code can be elaborated from its affine component $H_w(x, y, 1)$. There are $n = w^3$ affine points $p_j = (x_j, y_j, 1)$ $(1 \le j \le n)$ and a point at infinity $p_\infty = (0, 1, 0)$ [7]. Point p_∞ defines a pole basis Φ_w that consists of bivariate monomials $\phi_a = x^\delta y^\lambda$ $(0 \le \delta \le w, \lambda \ge 0)$ with increasing pole orders which are defined as

$$\upsilon_{p_\infty}(\phi_a^{-1}) = \upsilon_{p_\infty}((x^\delta y^\lambda)^{-1}) = w\delta + (w+1)\lambda. \tag{2}$$

978-1-4673-0223-4/12/$31.00 ©2012 IEEE

Consequently, the pole basis Φ_w is [7]:

$$\Phi_w = \{\phi_a(x,y) \mid \upsilon_{p_\infty}(\phi_a^{-1}) < \upsilon_{p_\infty}(\phi_{a+1}^{-1}),\ a \in \mathbb{N}\}, \quad (3)$$

where $\mathbb{N}$ is the set of nonnegative integers. E.g., $\Phi_2 = \{1, x, y, x^2, xy, y^2, x^2y, xy^2, y^3, \ldots\}$. Pole basis Φ_w collects all the bivariate monomials that define $\mathbb{F}_q[x,y]$ and $\mathbb{F}_q[x,y,z]$.

Since all the affine points can be distinguished by their x and y components, they can be simplified as $p_j = (x_j, y_j)$. Based on the affine points p_j and the pole basis Φ_w, the generator matrix $\mathbf{G} \in \mathbb{F}_q^{k\times n}$ of an (n,k) Hermitian code is defined as:

$$\mathbf{G} = \begin{pmatrix} \phi_0(p_1) & \phi_0(p_2) & \cdots & \phi_0(p_n) \\ \phi_1(p_1) & \phi_1(p_2) & \cdots & \phi_1(p_n) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{k-1}(p_1) & \phi_{k-1}(p_2) & \cdots & \phi_{k-1}(p_n) \end{pmatrix}, \quad (4)$$

where n and k are the length and dimension of the code, respectively. Given a message vector $\overline{F} = [F_1, F_2, \ldots, F_k] \in \mathbb{F}_q^k$, the codeword $\overline{C}$ can be generated by:

$$\overline{C} = [C_1, C_2, \ldots, C_n] = \overline{F} \cdot \mathbf{G}, \quad (5)$$

where $\overline{C} \in \mathbb{F}_q^n$. Vector $\overline{F}$ can be represented by a polynomial $F(x,y) = \sum_{j=1}^{k} F_j \phi_{j-1}$. The encoding process can be interpreted as evaluating the n affine points over the message polynomial. The length of Hermitian codes is $n = q^{3/2}$ which is larger than that of the RS code defined over $\mathbb{F}_q$. Its parity-check matrix $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$ is defined as:

$$\mathbf{H} = \begin{pmatrix} \phi_0(p_1) & \phi_0(p_2) & \cdots & \phi_0(p_n) \\ \phi_1(p_1) & \phi_1(p_2) & \cdots & \phi_1(p_n) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{n-k-1}(p_1) & \phi_{n-k-1}(p_2) & \cdots & \phi_{n-k-1}(p_n) \end{pmatrix}. \quad (6)$$

With a valid codeword $\overline{C}$, we have $\overline{C} \cdot \mathbf{H}^T = \mathbf{0}$ where $\mathbf{0}$ represents the all-zero vector resulting from the matrix product. In order to perform ABP decoding of Hermitian codes, the binary image of its parity-check matrix is required. Let $\sigma(x) \in \mathbb{F}_2[x]$ be a primitive polynomial of $\mathbb{F}_q$ and $\mathbf{A} \in \mathbb{F}_2^{\varpi\times\varpi}$ is the corresponding companion matrix. For any field element α^t $(t = 0, 1, 2, \ldots, q-2)$, mapping $\alpha^t \mapsto \mathbf{A}^t$ is applied. Consequently, the binary image of a parity-check matrix can be generated by replacing its entries α^t by their corresponding matrices $\mathbf{A}^t$. We use $\mathbf{H}_b$ to denote such a binary parity-check matrix and $\mathbf{H}_b \in \mathbb{F}_2^{(N-K)\times N}$, where $N = n\varpi$ and $K = k\varpi$. Let $\bar{c}$ denote the binary representation of codeword $\overline{C}$ as $\bar{c} = [c_1, c_2, \ldots, c_N]$, we have $\bar{c} \cdot \mathbf{H}_b^T = \mathbf{0}$.

III. Iterative Soft-Decision Decoding

The iterative decoding approach consists of two decoding stages. The first decoding stage is the ABP algorithm, which supplies the improved bit reliabilities. They are then converted into symbol reliabilities that are utilized by the second decoding stage, the KV algorithm. The improved bit reliabilities will also be given as feedback to the next round of ABP decoding process. The KV algorithm determines a list of output candidates $P(x,y)$ that are in the form of $F(x,y)$, and store them in the global list $\mathcal{L}$. In the end, the ML selection criterion is applied to $\mathcal{L}$ and pick out the candidate that has the minimal Euclidean distance to the received vector.

A. The ABP Decoding

The ABP algorithm will first perform Gaussian elimination on parity-check matrix $\mathbf{H}_b$ based on the bit reliability values. It yields an adapted parity-check matrix $\mathbf{H}_b'$, based on which the iterative BP algorithm is carried out.

It is assumed the channel is memoryless and $\bar{y} \in \mathbb{R}$ is the received vector observed from the channel. The log-likelihood ratio (LLR) of bit $c_\gamma (\gamma = 1, 2, \ldots, N)$ is determined by:

$$L(c_\gamma) = \ln \frac{\Pr[c_\gamma = 0|\bar{y}]}{\Pr[c_\gamma = 1|\bar{y}]}, \quad (7)$$

where $\Pr[c_\gamma = 0|\bar{y}]$ and $\Pr[c_\gamma = 1|\bar{y}]$ are the *a posteriori* probability (APP) values. The LLR vector $\overline{L}$ that collects all the LLR values of the coded bits is:

$$\overline{L} = [L(c_1), L(c_2), \ldots, L(c_{N-K}), \ldots, L(c_N)]. \quad (8)$$

With the magnitude $|L(c_\gamma)|$ being higher, bit c_γ is more reliable. Hence, all the magnitudes $|L(c_\gamma)|$ will be sorted in an ascending order, yielding a new bit index sequence $\gamma_1, \gamma_2, \ldots, \gamma_{N-K}, \ldots, \gamma_N$ that implies $|L(c_{\gamma_1})| < |L(c_{\gamma_2})| < \ldots < |L(c_{\gamma_{N-K}})| < \cdots < |L(c_{\gamma_N})|$. Consequently, the sorted LLR vector can be organized as:

$$\overline{L}_{srt} = [L(c_{\gamma_1}), L(c_{\gamma_2}), \ldots, L(c_{\gamma_{N-K}}), \cdots, L(c_{\gamma_N})]. \quad (9)$$

The bits that correspond to the first $N-K$ LLR values of $\overline{L}_{srt}$ are considered as the unreliable bits. Gaussian elimination will be performed on the columns w.r.t. the unreliable bits. Let Υ_γ denote the weight-1 column vector with 1 at its γth entry and 0 elsewhere. Gaussian elimination will first reduce column γ_1 to Υ_1, then reduce column γ_2 to Υ_2 and etc. It attempts to reduce the first $N-K$ independent columns implied by $\overline{L}_{srt}$ to the weight-1 columns. This process is called the matrix adaptation, resulting in an updated binary parity-check matrix $\mathbf{H}_b'$.

Let $h_{\beta\gamma}$ denote the entry of matrix $\mathbf{H}_b'$. The conventional BP algorithm will now be applied to $\mathbf{H}_b'$. Let us define

$$\mathbf{B}(\gamma) \triangleq \{\beta \mid h_{\beta\gamma} = 1,\ \forall\ h_{\beta\gamma} \in \mathbf{H}_b'\}, \quad (10)$$

$$\mathbf{\Gamma}(\beta) \triangleq \{\gamma \mid h_{\beta\gamma} = 1,\ \forall\ h_{\beta\gamma} \in \mathbf{H}_b'\}. \quad (11)$$

Let matrices $\mathbf{V}, \mathbf{U} \in \mathbb{R}^{(N-K)\times N}$ with entries $v_{\beta\gamma}$ and $u_{\beta\gamma}$, respectively. At the beginning, matrix $\mathbf{V}$ is initialized as:

$$v_{\beta\gamma} = L(c_\gamma) \cdot h_{\beta\gamma}, \forall\ 1 \le \beta \le N-K, 1 \le \gamma \le N. \quad (12)$$

First, the horizontal step will be performed as:

$$u_{\beta\gamma} = 2\tanh^{-1}\Big(\prod_{\tau\in\mathbf{\Gamma}(\beta)\setminus\gamma} \tanh\Big(\frac{v_{\beta\tau}}{2}\Big)\Big). \quad (13)$$

Then, the vertical step will be performed as:

$$v_{\beta\gamma} = L(c_\gamma) + \eta \sum_{\tau\in\mathbf{B}(\gamma)\setminus\beta} u_{\tau\gamma}, \quad (14)$$

where $0 < \eta \le 1$ is the damping factor [12] [13]. The extrinsic information of bit c_γ is given by:

$$L_{ext}(c_\gamma) = \sum_{\tau\in\mathbf{B}(\gamma)} u_{\tau\gamma}. \quad (15)$$

Calculations of (13)-(14) define one iteration of BP decoding. Let $\mathcal{N}_{\text{BP}}$ denote the predefined number of BP iterations. Once $\mathcal{N}_{\text{BP}}$ is reached, the LLR value of bit c_γ is updated by:

$$L'(c_\gamma) = L(c_\gamma) + \eta L_{ext}(c_\gamma). \tag{16}$$

As a result, the updated LLR vector $\overline{L}'$ can be formed as:

$$\overline{L}' = [L'(c_1), L'(c_2), \ldots, L'(c_{N-K}), \ldots, L'(c_N)]. \tag{17}$$

In the proposed ABP-KV algorithm, multiple matrix adaptations can be performed and each of them contains $\mathcal{N}_{\text{BP}}$ BP iterations. Given $\mathcal{N}_{\text{ADP}}$ as the number of matrix adaptations, the total number of BP iterations becomes $\mathcal{N}_{\text{ADP}}\mathcal{N}_{\text{BP}}$. If the next round of matrix adaptation is to be carried out, the LLR sorting process will be performed based on the updated LLR vector $\overline{L}'$. Through matrix adaptation, the density of the original parity-check matrix $\mathbf{H}_b$ is reduced and part of its short cycles are eliminated, making it more suitable for BP decoding. More importantly, since the columns w.r.t. the unreliable bits are reduced to weight-1, it prevents the propagation of the unreliable information during the BP decoding process.

B. The KV Decoding

After every $\mathcal{N}_{\text{BP}}$ BP iterations, each updated LLR value $L'(c_\gamma)$ will be converted back to a pair of bit APP values as:

$$\Pr[c_\gamma = 0|\bar{y}] = \frac{1}{1+e^{-L'(c_\gamma)}} \text{ and } \Pr[c_\gamma = 1|\bar{y}] = \frac{1}{1+e^{L'(c_\gamma)}}. \tag{18}$$

They are then utilized to generate the reliability matrix $\mathbf{\Pi}$ whose entries π_{ij} are symbol wise APP values defined as:

$$\pi_{ij} = \Pr[C_j = \rho_i \mid \bar{y}]. \tag{19}$$

Let Λ_i denote the binary representation of symbol ρ_i:

$$\Lambda_i = [\theta_1\ \theta_2\ \cdots\ \theta_\varpi \mid \rho_i = \sum_{\kappa=1}^{\varpi} \theta_\kappa \cdot \alpha^{\varpi-\kappa} \text{ and } \theta_\kappa \in [0,1]]. \tag{20}$$

For example, in $\mathbb{F}_8$, $\rho_7 = \alpha^2 + 1$ and $\Lambda_7 = [101]$. Hence, the symbol wise APP values π_{ij} can be determined by:

$$\pi_{ij} = \prod_{\kappa=1, \theta_\kappa \in \Lambda_i}^{\varpi} \Pr[c_{(j-1)\varpi+\kappa} = \theta_\kappa \mid \bar{y}]. \tag{21}$$

In general, every ϖ consecutive pairs of bit APP values will be multiplied as (21) in q different permutations, generating a column of matrix $\mathbf{\Pi}$. It will then be transferred into a multiplicity matrix $\mathbf{M}$ whose entries m_{ij} represent the interpolation multiplicity for points (p_j, ρ_i), where $1 \le i \le q$ and $1 \le j \le n$.

Interpolation will be performed based on $\mathbf{M}$, yielding an interpolated polynomial $Q \in \mathbb{F}_q[x, y, z]$ as:

$$Q(x, y, z) = \sum_{a,b \in \mathbb{N}} Q_{ab} \phi_a(x, y) z^b. \tag{22}$$

There are in total $\mathcal{C}(\mathbf{M}) = 0.5 \sum_{i,j} m_{ij}(m_{ij}+1)$ interpolation constraints [7] that polynomial Q needs to satisfy. Factorization will then be performed to find out the z-roots of Q [14]:

$$\{P(x, y) \mid Q(x, y, P(x, y)) = 0\}. \tag{23}$$

Let $\deg_w Q$ denote the weighted degree [7] of polynomial Q, factorization can produce at most $l = \lfloor \frac{\deg_w Q}{v_{p_\infty}(\phi_{k-1}^{-1})} \rfloor$ output candidates. Again, they will be stored in the global list $\mathcal{L}$.

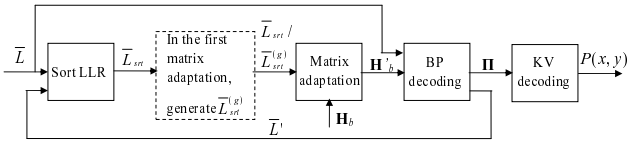

Fig. 1. Decoding parameters exchange of the ABP-KV algorithm.

C. Re-grouping of Unreliable Bits

The above description shows the ABP algorithm enables the columns w.r.t. the unreliable bits to be reduced to weight-1, and those bits are more likely to be corrected. However, it is possible that the reliable bits are wrongly estimated by their LLR values. Enabling their corresponding columns to be reduced can enhance the chance of them being corrected. Therefore, after the initial sorting process, vector $\overline{L}_{srt}$ can be restructured, creating different groups of bits whose corresponding columns will be reduced.

Let $\mathcal{N}_{\text{GR}}$ denote the designed number of groups of unreliable bits and $r = \lfloor N/\mathcal{N}_{\text{GR}} \rfloor$. The original sorted LLR vector $\overline{L}_{srt}$ can be expressed as [13]:

$$\begin{aligned} \overline{L}_{srt} &= [L(c_{\gamma_1}), \ldots, L(c_{\gamma_r}), L(c_{\gamma_{r+1}}), \ldots, L(c_{\gamma_{2r}}), \ldots, \\ &\quad L(c_{\gamma_{(g-1)r+1}}), \ldots, L(c_{\gamma_{gr}}), \ldots, L(c_{\gamma_N})], \end{aligned} \tag{24}$$

where $1 \le g \le \mathcal{N}_{\text{GR}}$ is the group index. $\overline{L}_{srt}^{(g)}$ is used to denote the restructured LLR vector of group g. For group 1, $\overline{L}_{srt}^{(1)} = \overline{L}_{srt}$ since no restructuring is needed. While for group g with $g > 1$, vector $\overline{L}_{srt}$ will be restructured to:

$$\begin{aligned} \overline{L}_{srt}^{(g)} &= [L(c_{\gamma_{(g-1)r+1}}), \ldots, L(c_{\gamma_{gr}}), L(c_{\gamma_1}), \ldots, \\ &\quad L(c_{\gamma_{(g-1)r}}), L(c_{\gamma_{gr+1}}), \ldots, L(c_{\gamma_N})]. \end{aligned} \tag{25}$$

Again, matrix adaptation will be performed on the first $N-K$ independent columns implied by $\overline{L}_{srt}^{(g)}$.

Given $\mathcal{N}_{\text{ADP}}$ matrix adaptations, the LLR vector restructuring process should only be performed prior to the first matrix adaptation of the ABP algorithm. Hence, the first adaptation is performed based on $\overline{L}_{srt}^{(g)}$, while the rest of the matrix adaptations are performed based on the updated LLR vector $\overline{L}'$. Generalizing this section, Fig. 1 illustrates the decoding parameters exchange between different steps of the ABP-KV algorithm. Notice that with $\mathcal{N}_{\text{GR}}$ unreliable groups, the ABP-KV decoding process will be deployed $\mathcal{N}_{\text{GR}}$ times, each of which inherits a specific restructuring pattern for $\overline{L}_{srt}^{(g)}$. Summarizing this section, the proposed ABP-KV decoding algorithm is presented as Algorithm 1.

IV. Geometric Analysis of ABP Decoding

The conventional BP decoding can be seen as a gradient descent decoding problem [12] [15]. The coded bit LLR values $L(c_\gamma)$ can be normalized to the region of $[-1, +1]$ by the following mapping function:

$$\xi(L(c_\gamma)) = \tanh\left(\frac{L(c_\gamma)}{2}\right) = \frac{e^{L(c_\gamma)} - 1}{e^{L(c_\gamma)} + 1}. \tag{26}$$

Algorithm 1 ABP-KV decoding of Hermitian codes

for each group g **do**
  Let $\overline{L}' = \overline{L}$;
  for each parity-check matrix adaptation **do**
    Generate a sorted LLR vector $\overline{L}_{srt}$ based on $\overline{L}'$;
    if it is the first matrix adaptation **then**
      Restructure the sorted LLR vector to $\overline{L}_{srt}^{(g)}$;
      Generate matrix $\mathbf{H}_b'$ based on $\overline{L}_{srt}^{(g)}$.
    else
      Generate matrix $\mathbf{H}_b'$ based on $\overline{L}_{srt}$;
    end if
    Initialize matrix $\mathbf{V}$ as in (12);
    for each BP iteration **do**
      Perform the horizontal step as in (13);
      Perform the vertical step as in (14);
    end for
    Determine the extrinsic information of bit c_γ as in (15) and update its LLR value as in (16);
    Form the updated LLR vector $\overline{L}'$ as in (17);
    Generate N pairs of bit APP values as in (18);
    Determine the reliability matrix $\mathbf{\Pi}$ as in (21);
    Transfer matrix $\mathbf{\Pi}$ into matrix $\mathbf{M}$;
    Perform interpolation to determine Q of (22);
    Perform factorization to find out $P(x, y)$ of (23);
  end for
end for

Again, with the magnitude $|\xi(L(c_\gamma))|$ being higher, bit c_γ is more reliable. By normalizing all the LLR values of a codeword as in (26), we can form vector $\overline{T}$ as:

$$\overline{T} = [T_1, T_2, \ldots, T_N] = [\xi(L(c_1)), \xi(L(c_2)), \ldots, \xi(L(c_N))]. \quad (27)$$

It corresponds to an estimated codeword that satisfies all the checks. With matrix $\mathbf{H}_b'$ and vector $\overline{T}$, the potential function $\mathcal{P}(\mathbf{H}_b', \overline{T})$ of a Hermitian code can be defined as [12] [15]:

$$\mathcal{P}(\mathbf{H}_b', \overline{T}) = -\sum_{\beta=1}^{N-K} \prod_{\gamma \in \mathbf{\Gamma}(\beta)} T_\gamma. \quad (28)$$

The quantization of $\mathcal{P}(\mathbf{H}_b', \overline{T})$ describes the reliability of vector $\overline{T}$. Consequently, the LLR updates of (16) can be seen as the gradient descent update as follows:

$$T_\gamma' = T_\gamma - \eta \frac{\partial \mathcal{P}(\mathbf{H}_b', \overline{T})}{\partial T_\gamma} = T_\gamma + \eta \Big(\sum_{\beta=1}^{N-K} \prod_{\tau \in \mathbf{\Gamma}(\beta) \setminus \gamma} T_\tau \Big). \quad (29)$$

With all the checks of matrix $\mathbf{H}_b'$ being satisfied, a valid codeword is reached if $|T_\gamma| = 1$ for $\gamma = 1, 2, \ldots, N$. Consequently, $\min\{\mathcal{P}(\mathbf{H}_b', \overline{T})\} = -(N - K)$. Therefore, finding an estimated codeword using the BP algorithm can be interpreted as identifying the vertex at which the potential function is minimized.

Without performing matrix adaptation, the gradient descent decoding is easily hindered at some pseudo-equilibrium points which prevent the potential function from reaching its minimum. Fig. 2 shows the convergence behavior of the potential function of the (64, 47) Hermitian code, which is measured in the additive white Gaussian noise (AWGN) channel. It shows without matrix adaptation, the potential function cannot converge to its minimum. By increasing the signal-to-noise ratio (SNR), a valid codeword is reached with fewer iterations. However, the matrix adaptation can only reduce the density of $\mathbf{H}_b'$ to around 37%, which is not sparse enough for the BP algorithm to produce a reliable error-correction performance. Without making a hard-decision on its decoding outputs, the ABP algorithm will have to be integrated by a conventional algebraic decoding algorithm. In order to fully utilize its soft output, the KV algorithm becomes an obvious choice.

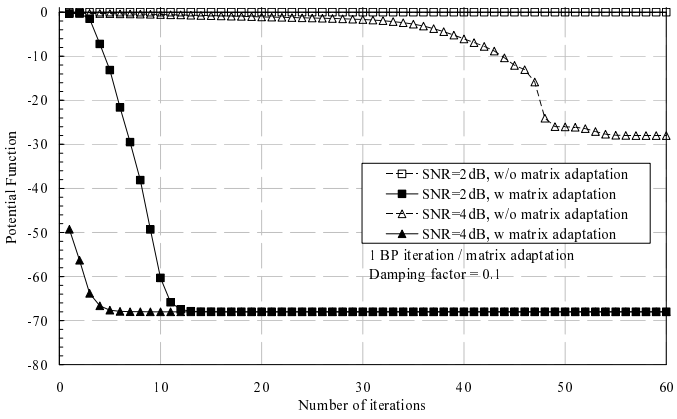

Fig. 2. Convergence of the potential function of the (64, 47) Hermitian code.

V. Performance Evaluations

The section presents the ABP-KV decoding performance for Hermitian codes. The results are obtained in the AWGN channel using BPSK modulation. The ABP-KV algorithm is parameterized by the ternary tuple $(\mathcal{N}_{\text{GR}}, \mathcal{N}_{\text{ADP}}, \mathcal{N}_{\text{BP}})$. It implies the algorithm will perform $\mathcal{N}_{\text{GR}}\mathcal{N}_{\text{ADP}}$ Gaussian eliminations and KV decodings, and $\mathcal{N}_{\text{GR}}\mathcal{N}_{\text{ADP}}\mathcal{N}_{\text{BP}}$ BP iterations. The KV algorithm is parameterized by the interpolation cost $\mathcal{C}$ that defines the scale of list decoding complexity.

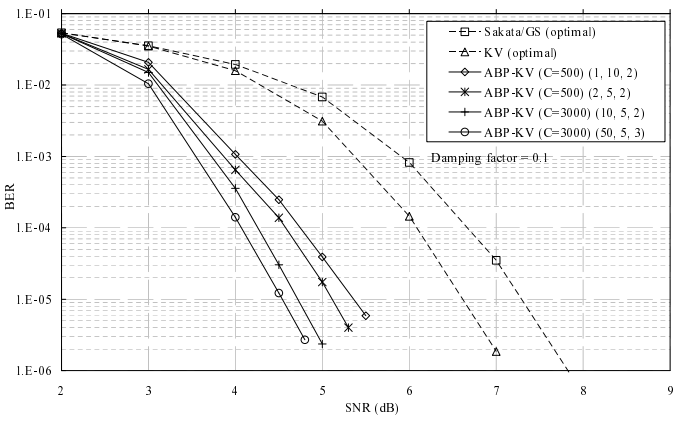

Fig. 3. Performance of ABP-KV decoding of (64, 52) Hermitian code.

Fig. 3 shows the ABP-KV decoding performance of the (64, 52) Hermitian code. The performances of the Sakata

algorithm and the optimal GS and KV algorithms are shown as comparison benchmarks. It can be seen that ABP-KV decoding achieves significant performance improvements over the existing decoding algorithms. Notice that ABP-KV decoding with $(\mathcal{C} = 500)(2, 5, 2)$ outperforms that with $(\mathcal{C} = 500)(1, 10, 2)$. However, these two sets of parameters indicate a similar decoding complexity. It implies that given a budget on the amount of Gaussian eliminations, it is beneficial to deploy them among different unreliable groups. Further performance improvements can be made by increasing the decoding parameters, such as the results associated with parameters $(\mathcal{C} = 3000)(10, 5, 2)$ and $(\mathcal{C} = 3000)(50, 5, 3)$.

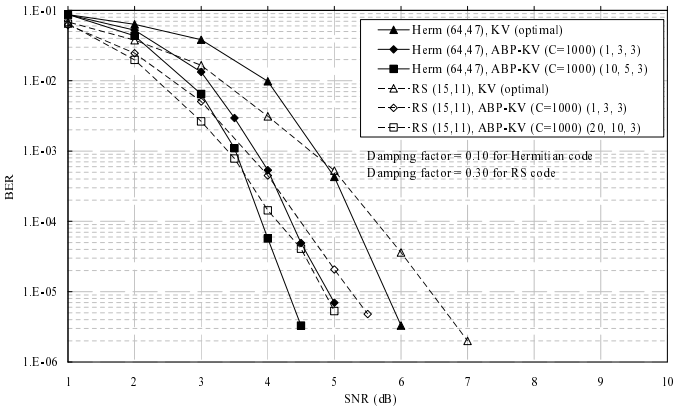

Fig. 4. Performance comparison of the Hermitian code and the RS code.

Fig. 4 compares the ABP-KV decoding performance between the (64, 47) Hermitian code and the (15, 11) RS code. They are defined over $\mathbb{F}_{16}$ and have a similar code rate. It can be seen that with the same decoding parameter, i.e., $(\mathcal{C} = 1000)(1, 3, 3)$, the Hermitian code prevails. Such a performance advantage is due to the size of the Hermitian code. Its size is about 4 times of the RS code. Since the decoding complexity is proportional to the size of the code, ABP-KV decoding of the Hermitian code would be more complex. In order to compare the performance based on a similar decoding complexity, we further compare ABP-KV decoding of the Hermitian code with $(\mathcal{C} = 1000)(10, 5, 3)$ and the RS code with $(\mathcal{C} = 1000)(20, 10, 3)$. For the RS code, the numbers of Gaussian eliminations, KV decodings and BP iterations are 4 times of those for the Hermitian code. Such a setting compensates the complexity advantage of the RS code due to its shorter length. However, the Hermitian code still outperforms the RS code with 0.5dB coding gain at bit error rate (BER) of 10^{-5}. Hence, with a similar decoding complexity, the Hermitian code still has a performance advantage.

VI. Conclusions

This paper has proposed the iterative ABP-KV decoding algorithm for Hermitian codes. With the ABP algorithm to improve the reliability of the received information, the KV algorithm performs the polynomial-time list decoding to retrieve the intended message. Re-grouping of the unreliable bits that triggers a different ABP decoding process has been introduced to enhance the error-correction performance. Geometric analysis of the ABP algorithm has been presented, demonstrating the necessity of performing parity-check matrix adaption and integrating the ABP and KV algorithms. Our performance evaluations show that the proposed algorithm is so far the most advanced decoding approach for Hermitian codes. With a similar decoding complexity, ABP-KV decoding of Hermitian code outperforms that of RS code. Therefore, the proposed iterative decoding scheme can be considered for a wider range of applications.

Acknowledgements

This research is supported by the National Natural Science Foundation of China (NSFC) with project ID 61001094, the Guangdong Natural Science Foundation (GDNSF) with project ID 10451027501005078 and the National Basic Research Program of China (973 Program) with project ID 2012CB316100.

References

[1] M. Johnston and R. Carrasco, "Construction and performance of algebraic-geometric codes over AWGN and fading channels," *IEE Proc. Commun.*, vol. 152, pp. 713-722, 2005.

[2] L. Chen, R. Carrasco and M. Johnston, "Soft-decision list decoding of Hermitian codes," *IEEE Trans. Commun.*, vol. 57 (8), pp. 2169-2176, Aug. 2009.

[3] S. Sakata, J. Justesen, Y. Madelung, H. Jensen, and T. Hoholdt, "Fast decoding of algebraic-geometric codes up to the designed minimum distance," *IEEE Trans. Inform. Theory*, vol. 41, pp. 1672-1677, Nov. 1995.

[4] G. Feng and T. Rao, "Decoding algebraic-geometric codes up to the designed minimum distance," *IEEE Trans. Inform. Theory*, vol. 39, pp. 37-46, Jan. 1993.

[5] V. Guruswami and M. Sudan, "Improved decoding of Reed-Solomon and algebraic-geometric codes," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1757-1767, Sept. 1999.

[6] T. Hoholdt and R. Nielsen, "Decoding Hermitian codes with Sudan's algorithm," in *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (Lecture Notes in Computer Science)*, vol. 1719/1999, Springer-Verlag, pp. 260-270, 1999.

[7] L. Chen, R. Carrasco and M. Johnston, "Reduced complexity for list decoding Hermitian codes," *IEEE Trans. Wireless Commun.*, vol. 7 (11), pp. 4353-4361, Nov. 2008.

[8] K. Lee and M. O'Sullivan, "Algebraic soft-decision decoding of Hermitian codes," *IEEE Trans. Inform. Theory*, vol. 56 (6). pp. 2587-2600, Jun. 2010.

[9] V. Ponnampalam and B. Vucetic, "Soft deisioin decoding of Reed-Solomon codes," *IEEE Trans. Commun.*, vol. 50 (11), pp. 1758-1768, Nov. 2002.

[10] M. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inform. Theory*, vol. 41 (5), pp. 1379-1396, Sept. 1995.

[11] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes," *IEEE Trans. Inform. Theory*, vol. 49 (11), pp. 2809-2825, Nov. 2003.

[12] J. Jiang and K. Narayanan, "Iterative soft-input-soft-output decoding of Reed-Solomon codes by adapting the parity check matrix," *IEEE Trans. Inform. Theory*, vol. 52 (8), pp. 3746-3756, Aug. 2006.

[13] M. El-Khamy and R. McEliece, "Iterative algebraic soft-decision list decoding of Reed-Solomon codes," *IEEE J. Sel. Areas in Commun.*, vol.24 (3), pp. 481-490, Mar. 2006.

[14] X. W. Wu and P. Siegel, "Efficient root-finding algorithm with application to list decoding of algebraic-geometric codes," *IEEE Trans. Inform. Theory*, vol. 47 (6), pp. 2579-2587, Sept. 2001.

[15] R. Lucas, M. Bossert and M. Breitbach, "On iterative soft-decision decoding of linear binary block codes and product codes," *IEEE Jour. Sel. Areas in Comm.*, vol. 16 (2), pp. 276-296, Feb. 1998.