Turbo-like iterative least-squares decoding of analogue codes

F. Hu and W. Henkel

A turbo-like iterative decoding scheme for analogue product codes is described. It is proved that the iterative decoding method is an iterative projection in Euclidean space and converges to the least-squares solution. Using this geometric point of view, any block analogue codes can be decoded by a similar iterative method. The described procedure may serve as a step towards a more intuitive understanding of turbo decoding.

Encoding: We first study an analogue product code with 'parity-check' component codes [1]. This transfers the construction of a product code with binary parity check component codes to the analogue (real/complex) number field. $K = k^2$ analogue symbols are first arranged as a $k \times k$ matrix and then mapped into a $(k+1) \times (k+1)$ matrix $\boldsymbol{X}$ such that the rows and columns add up to zero.

To simplify the analysis, we define $\boldsymbol{x}$ to contain the sequence of columns of $\boldsymbol{X}$:

$$\boldsymbol{x} = \mathrm{vec}(\boldsymbol{X}) \tag{1}$$

A parity-check matrix $\boldsymbol{H}$ with $\boldsymbol{Hx} = \boldsymbol{0}$ can be constructed as

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{H}_1 \\ \boldsymbol{H}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} \otimes \boldsymbol{1}^T \\ \boldsymbol{1}^T \otimes \boldsymbol{I} \end{bmatrix} \tag{2}$$

where $\otimes$ denotes the Kronecker product and 1 is a $(k+1)$-dimensional column vector of 1s. $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$ correspond to column and row constraints, respectively. It is known that the codeword space χ is uniquely defined by the parity-check matrix as

$$\chi = \{\boldsymbol{x} : \boldsymbol{Hx} = \boldsymbol{0}\} \tag{3}$$

If we define two spaces corresponding to the column and row constraints as

$$\begin{aligned} \mathcal{H}_1 &= \{\boldsymbol{x} : \boldsymbol{H}_1\boldsymbol{x} = \boldsymbol{0}\} \\ \mathcal{H}_2 &= \{\boldsymbol{x} : \boldsymbol{H}_2\boldsymbol{x} = \boldsymbol{0}\} \end{aligned} \tag{4}$$

then $\chi = \mathcal{H}_1 \cap \mathcal{H}_2$.

Decoding: Given a disturbed received word $\boldsymbol{Y}$ and knowing that the rows and columns sum to zeros, we compute two types of extrinsic information for each symbol:

$$\begin{aligned} y_{1i,j} &= -\sum_{j=1, j\neq 1}^{k+1} y_{i,j} \quad i,j = 1,\ldots,k+1 \\ y_{2i,j} &= -\sum_{i=1, j\neq 1} y_{i,j} \quad i,j = 1,\ldots,k+1 \end{aligned} \tag{5}$$

where $y_{i,j}$ denotes the components of $\boldsymbol{Y}$. The algorithm is the analogue counterpart of the one for binary product codes usually used to explain turbo decoding. Equations (5) show the extrinsic information of the rows and columns, respectively.

Letting $\boldsymbol{y} = \mathrm{vec}(\boldsymbol{Y})$, the two types of extrinsic information can also be represented in vector format:

$$(\boldsymbol{I} \otimes \bar{\boldsymbol{I}})\boldsymbol{y} \quad \text{and} \quad (\bar{\boldsymbol{I}} \otimes \boldsymbol{I})\boldsymbol{y} \tag{6}$$

where $\boldsymbol{I}$ is the $(k+1) \times (k+1)$ identity matrix, $\bar{\boldsymbol{I}} = \boldsymbol{E} - \boldsymbol{I}$, $\boldsymbol{E}$ is a $(k+1) \times (k+1)$ matrix of 1s.

The decoding method is an analogue counterpart of the turbo decoding for binary codes in [2] where the estimated vector is computed as the weighted sum of the previous vector and the extrinsic information vectors:

$$\begin{aligned} \boldsymbol{y}_{(v)} &= \frac{1}{1+2\omega}(y_{(v-1)} - \omega(\boldsymbol{I} \otimes \bar{\boldsymbol{I}})y_{(v-1)} - \omega(\bar{\boldsymbol{I}} \otimes \boldsymbol{I})\boldsymbol{y}_{(v-1)}) \\ &= \boldsymbol{y}_{(v-1)} - \boldsymbol{y}^{(1)}_{(v-1)} - \boldsymbol{y}^{(2)}_{(v-1)} \end{aligned} \tag{7}$$

ω is the weight parameter, v denotes the iteration step index, and

$$\begin{aligned} \boldsymbol{y}^{(1)}_{(v-1)} &= \frac{\omega}{1+2\omega}(\boldsymbol{I} \otimes \boldsymbol{E})\boldsymbol{y}_{(v-1)} \\ \boldsymbol{y}^{(2)}_{(v-1)} &= \frac{\omega}{1+2\omega}(\boldsymbol{E} \otimes \boldsymbol{I})\boldsymbol{y}_{(v-1)} \end{aligned} \tag{8}$$

We can prove that $\boldsymbol{y}^{(1)}_{(v-1)}$ is orthogonal to space $\mathcal{H}_1$ by observing that it is a linear combination of columns of $\boldsymbol{H}_1^T$, i.e.

$$\boldsymbol{y}^{(1)}_{(v-1)} = \boldsymbol{H}_1^T \alpha \tag{9}$$

where

$$\alpha = \frac{\omega}{1+2\omega}\boldsymbol{H}_1\boldsymbol{y}_{(v-1)} \tag{10}$$

Furthermore, when $\omega = 1/k - 1$, $\boldsymbol{H}_1 \cdot (\boldsymbol{y}_{(v-1)} - \boldsymbol{y}^{(1)}_{(v-1)}) = 0$. This means $\boldsymbol{y}^{(1)}_{(v-1)}$ lies in $\mathcal{H}_1$ by the definition of $\mathcal{H}_1$ (see (4)). Thus, when $w = 1/k - 1$, $\boldsymbol{y}_{(v-1)} - \boldsymbol{y}^{(1)}_{(v-1)}$ is the projection of $\boldsymbol{y}_{(v-1)}$ onto space $\mathcal{H}_1$. Similarly, $\boldsymbol{y}_{(v-1)} - \boldsymbol{y}^{(2)}_{(v-1)}$ is the projection of $\boldsymbol{y}_{(v-1)}$ onto space $\mathcal{H}_2$, see Fig. 1.

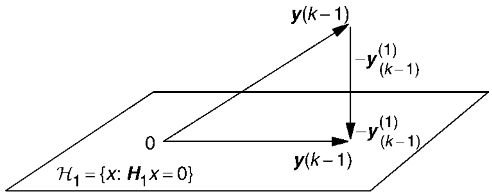

Fig. 1 *Projection of $\boldsymbol{y}_{(v-1)}$ onto space $\mathcal{H}_1$*

A geometric illustration of (7) for $\omega = 1/k - 1$ is given in Fig. 2. At each iteration step v, the previous vector $\boldsymbol{y}_{(v-1)}$ is projected onto $\mathcal{H}_1$ and $\mathcal{H}_2$ in parallel, which delivers two projection vectors $-\boldsymbol{y}^{(1)}_{(v-1)}$ and $-\boldsymbol{y}^{(2)}_{(v-1)}$. Both projection vectors are added to $\boldsymbol{y}_{(v-1)}$ resulting in the current vector $\boldsymbol{y}_{(v)}$. From Fig. 2 we see that this process makes $\boldsymbol{y}_{(\infty)}$ converge to $\mathcal{H}_1 \cap \mathcal{H}_2$ which is actually the least-squares solution. It is shown in (8) that the length of $\boldsymbol{y}^{(1)}_{(v-1)}$, $\boldsymbol{y}^{(2)}_{(v-1)}$ is determined by the parameter ω; the larger ω, the longer the length. To ensure $\boldsymbol{y}_{(\infty)} \in \mathcal{H}_1 \cap \mathcal{H}_2$, $\boldsymbol{y}_{(v-1)}$ should always be located in between $\mathcal{H}_1$ and $\mathcal{H}_2$ in each iteration step which requires $0 < \omega < 1/(k-1)$. Furthermore, we notice that the speed of the convergence can be improved if w is properly chosen.

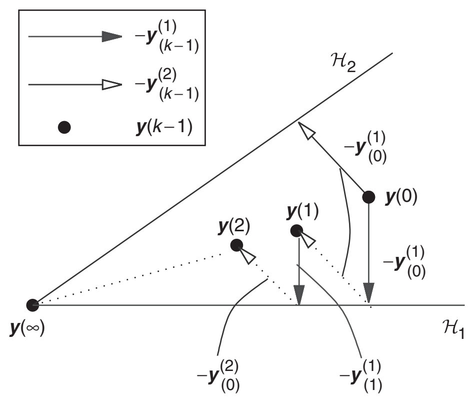

Fig. 2 *Geometric illustration of iterative decoding when $\omega = 1/k - 1$*

Iterative decoding for any analogue block code: Using this geometric point of view, any arbitrary analogue block codes can be decoded in the same iterative method by splitting the parity-check matrix. Let the parity-check matrix $\boldsymbol{H}$ be split as

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{H}_1 \\ \boldsymbol{H}_2 \end{bmatrix} \tag{11}$$

Let $\mathcal{H}_1$, $\mathcal{H}_2$ be spaces orthogonal to $\boldsymbol{H}_1$, $\boldsymbol{H}_2$, respectively. Our decoding approach is to find the projections of a vector $\boldsymbol{y}_{(v-1)}$ onto $\mathcal{H}_1$, $\mathcal{H}_2$ at each iteration step v.

Assume that $\boldsymbol{y}_{(v-1)} - \boldsymbol{y}^{(1)}_{(v-1)}$ is the projection of $\boldsymbol{y}_{(v-1)}$ onto space $\mathcal{H}_1$ and $\boldsymbol{y}_{(v-1)} - \boldsymbol{y}^{(2)}_{(v-1)}$ is the projection of $\boldsymbol{y}_{(v-1)}$ onto space $\mathcal{H}_2$. The iterative algorithm can be written as

$$\boldsymbol{y}_{(v)} = \boldsymbol{y}_{(v-1)} - \lambda \boldsymbol{y}^{(1)}_{(v-1)} - \lambda \boldsymbol{y}^{(2)}_{(v-2)} \qquad (12)$$

As long as $\lambda < 1$, $\boldsymbol{y}_{(\infty)}$ will converge to the least-squares solution.

According to Fig. 1, $\boldsymbol{y}^{(1)}_{(v-1)}$ is orthogonal to $\mathcal{H}_1$, thus can be expressed as a linear combination of columns of $\mathcal{H}_1^T$. Without loss of generality, suppose

$$\boldsymbol{y}^{(1)}_{(v-1)} = \boldsymbol{H}_1^T \alpha \qquad (13)$$

Since $\boldsymbol{y}_{(v-1)} - \boldsymbol{y}^{(1)}_{(v-1)}$ lies in $\mathcal{H}_1$, we have

$$\boldsymbol{H}_1 \cdot (\boldsymbol{y}_{(v-1)} - \boldsymbol{y}^{(1)}_{(v-1)}) = \boldsymbol{H}_1 \cdot (\boldsymbol{y}_{(v-1)} - \boldsymbol{H}_1^T \alpha) = 0 \qquad (14)$$

The solution for α is

$$\alpha = (\boldsymbol{H}_1 \boldsymbol{H}_1^T)^{-1} \boldsymbol{H}_1 \boldsymbol{y}_{(v-1)} \qquad (15)$$

The prerequisite for this solution is that $\boldsymbol{H}_1$ is a row full-rank matrix, otherwise the inverse of $(\boldsymbol{H}_1 \boldsymbol{H}_1^T)$ does not exist. Now, $\boldsymbol{y}^{(1)}_{(v-1)}$ can be expressed as

$$\boldsymbol{y}^{(1)}_{(v-1)} = \boldsymbol{H}_1^T (\boldsymbol{H}_1 \boldsymbol{H}_1^T)^{-1} \boldsymbol{H}_1 \boldsymbol{y}_{(v-1)} \qquad (16)$$

similarly,

$$\boldsymbol{y}^{(2)}_{(v-1)} = \boldsymbol{H}_2^T (\boldsymbol{H}_2 \boldsymbol{H}_2^T)^{-1} \boldsymbol{H}_2 \boldsymbol{y}_{(v-1)} \qquad (17)$$

where $\boldsymbol{H}_2$ must also be a row full-rank matrix.

Conclusion: Starting from a geometric illustration of the iterative decoding of analogue product codes, we have found an iterative least-squares decoding procedure for an arbitrary linear analogue block code by splitting its parity-check matrix in two and projecting received codewords onto the null spaces of these two matrices in an iterative fashion.

This work is part of the FP6/IST project M-Pipe and is co-funded by the European Commission.

11 August 2005
Electronics Letters online no: 20052912
doi: 10.1049/el:20052912

F. Hu and W. Henkel (*International University Bremen, Campus Ring 1, D-28759 Bremen, Germany*)

E-mail: w.henkel@iu-bremen.de

References

1 Mura, M., Henkel, W., and Cottatellucci, L.: 'Iterative least-squares decoding of analog product codes'. Proc. IEEE Int. Symp. on Information Theory (ISIT 2003), Yokohama, Japan, June–July 2003

2 Hagenauer, J., Offer, E., and Papke, L.: 'Iterative decoding of binary block and convolutional codes', *IEEE Trans. Inf. Theory*, 1996, **42**, (2), pp. 429–445