Iterative Least-Squares Decoding of Analog Product Codes

Marzia Mura
Telecommunications Research Center (ftw.)
currently with
Abbeynet srl
I-09028 Sestu
E-mail: marzia.mura@abbeynet.it

Werner Henkel
University of Applied Sciences Bremen
Neustadtswall 30
D-28199 Bremen
(formerly ftw.)
E-mail: werner.henkel@ieee.org
URL: http://trsys.fbe.hs-bremen.de

Laura Cottatellucci
Telecommunications Research Center (ftw.)
Donau-City-Str. 1
A-1220 Vienna, Austria
E-mail: cottatellucci@ftw.at
URL: http://www.ftw.at

Abstract — **We prove that a Turbo-like iterative decoding of an analog product code with parity-check component codes lead to the least-squares solution.**

I. Encoding

Analog codes date back to early papers by Jack Wolf in 1983 (see also own earlier work at trsys.fbe.hs-bremen.de), where he described the decoding of Reed-Solomon codes over complex numbers with a focus on impulse noise. As a first step into iterative decoding of analog codes, in here, we define analog product codes with analog parity-check component codes in rows and columns. Under analog parity-check code we understand appending the negative disparity (sum) of the information components. Thus, every row and every column will be DC-free. We obtain the code array

$$X^* = \left[\begin{array}{ccc|c} x_{1,1} & \cdots & x_{1,n} & x_{1,n+1} \\ \vdots & \ddots & \vdots & \vdots \\ x_{n,1} & \cdots & x_{n,n} & x_{n,n+1} \\ \hline x_{n+1,1} & \cdots & x_{n+1,n} & x_{n+1,n+1} \end{array}\right], \qquad (1)$$

with

$$x_{j,n+1} = -\sum_{i=1}^{n} x_{j,i}$$
$$x_{n+1,j} = -\sum_{i=1}^{n} x_{i,j}$$
$$x_{n+1,n+1} = \sum_{i=1}^{n}\sum_{j=1}^{n} x_{i,j} \,.$$

II. Decoding

Given the received codeword $R_{(0)}$, the basis idea is to compute two new matrices R_1 and R_2, whose elements are

$$\begin{aligned} r_{1j,i} &= -\sum_{\substack{i=1\\ i\neq j}}^{n+1} r_{j,i}\,, \qquad j = 1,\cdots,n+1\,, \\ r_{2j,i} &= -\sum_{\substack{j=1\\ j\neq i}}^{n+1} r_{j,i}\,, \qquad i = 1,\cdots,n+1\,, \end{aligned} \qquad (2)$$

with $r_{j,i}$ denoting the components of $R_{(k)}$, $k = 0, 1, \ldots$. The algorithm is the analog counterpart of the one for binary array codes usually used to explain Turbo decoding. Equations (2) show the extrinsic information of the rows and columns, respectively, knowing that the rows and columns sum to zero.

The steps in (2) can be written as the elements of two update matrices

$$\bar{I}R_{(n-1)} \text{ and } \bar{I}R^{T}_{(n-1)}\,, \qquad (3)$$

where $\bar{I}$ is an identity matrix with zeros replaced by ones and vice versa. The new matrix is then computed as the weighted sum of the previous matrix and the update matrices:

$$\begin{aligned} R_{(n)} &= (R_{(n-1)} - w\bar{I}R_{(n-1)} - w\bar{I}R^{T}_{(n-1)})/(1+2w) \\ &= (R_{(n-1)} - w\bar{I}(R_{(n-1)} + R^{T}_{(n-1)}))/(1+2w) \end{aligned} \qquad (4)$$

To study the convergency properties, the received matrix is written as a vector $y_{(n)}$ that contains the sequence of its rows. The computation can then be described as

$$\begin{aligned} y_{(k)} &= \Phi y_{(k-1)} \\ &= \Phi^k y_{(0)}\,. \end{aligned} \qquad (5)$$

$y_{(k)}$ is the vector obtained at the k^{th} iteration step. Φ is the iteration matrix of dimension $(n+1)^2 \times (n+1)^2$ defined as

$$\begin{aligned} \Phi &= (I - wM_1 - wM_2)/(1+2w) \\ &= (I - w(M_1 + M_2))/(1+2w)\,. \end{aligned} \qquad (6)$$

I is the $(n+1) \times (n+1)$ identity matrix, M_1 and M_2 describe the computation per columns and per rows, respectively, w is a weight which depends on the matrix dimension.

The eigenvalues of the operator Φ have been found to be

$$\begin{array}{lll} 1 & \text{eigenvalue equal to} & \frac{1-2wn}{1+2w}\,, \\ 2n & \text{eigenvalues equal to} & \frac{1-w(n-1)}{1+2w}\,, \\ n2 & \text{eigenvalues equal to} & 1\,. \end{array} \qquad (7)$$

Convergence will be obtained, if all eigenvalues are less or equal to one. The eigenvalues that are equal to one correspond to the eigenvectors that span the solution space. If the others are forced to below one by choosing the weight to be

$$0 < w < 1/n\,, \qquad (8)$$

convergence will be obtained.

Since eigenvectors associated with different eigenvalues are mutually orthogonal, the iterative algorithm actually leads to the least squares solution, i.e., it projects onto the solution subspace.

Acknowledgments

We would like to thank Markus Rupp for some early contributions regarding the convergence conditions and Jossy Sayir for valuable discussions and critical reading of the manuscript.