Least Squares Solution for Error Correction on the Real Field Using Quantized DFT Codes

Mojtaba Vaezi and Fabrice Labeau

McGill University

European Signal Processing Conference (EUSIPCO)
Bucharest, Romania

August 31, 2012

Outline

Real BCH-DFT Codes

Applications

Motivations for studying BCH-DFT codes

- Resilience to additive noise including quantization error
- Erasures and errors correction (channel coding)
- Distributed lossy source coding (new)
- Better performance w.r.t. delay and complexity
- Better performance under particular channel characteristics

Real BCH-DFT Codes

Applications

Motivations for studying BCH-DFT codes

- Resilience to additive noise including quantization error
- Erasures and errors correction (channel coding)
- Distributed lossy source coding (new)
- Better performance w.r.t. delay and complexity
- Better performance under particular channel characteristics

Real BCH-DFT Codes

Applications

Motivations for studying BCH-DFT codes

- Resilience to additive noise including quantization error
- Erasures and errors correction (channel coding)
- Distributed lossy source coding (new)
- Better performance w.r.t. delay and complexity
- Better performance under particular channel characteristics

Connection to Frame Theory

- Complex BCH-DFT codes are harmonic frames
- Real BCH-DFT codes are rotated harmonic frames

Real BCH-DFT Codes

Encoding

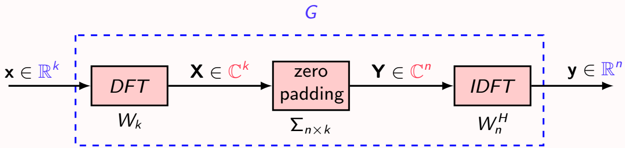

Figure: Real BCH-DFT encoding scheme

$$G = \sqrt{\frac{n}{k}} W_n^H \Sigma W_k$$

- $\Sigma_{n \times k}$ inserts n-k consecutive zeros in the transform domain $\Longrightarrow$ BCH code
- DFT is used to convert vector $\mathbf{x} \in \mathbb{R}^k$ to a circularly symmetric $\mathbf{X} \in \mathbb{C}^k$, guaranteeing a real $\mathbf{y}$
- Removing the DFT block, we obtain complex BCH-DFT codes

Real BCH-DFT Codes

Coding scheme

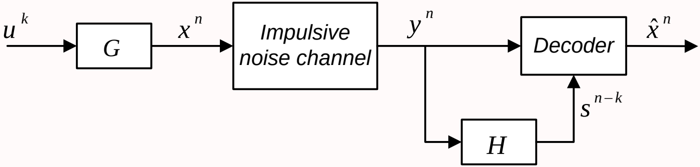

Figure: Channel coding using real-valued BCH codes

- H takes N-K columns of W_N^H corresponding to zeros of Σ
- For every codeword, $s = Hy = HGx \equiv 0$

Real BCH-DFT Codes

Coding scheme

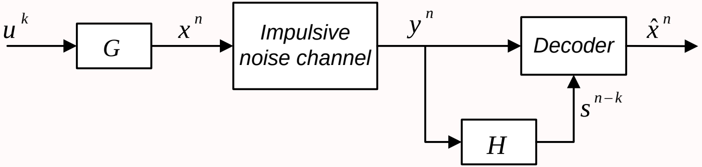

Figure: Channel coding using real-valued BCH codes

- H takes N-K columns of W_N^H corresponding to zeros of Σ
- For every codeword, $s = Hy = HGx \equiv 0$

Without quantization:

$y^n = x^n + e^n \Rightarrow s_y = s_e$

Real BCH-DFT codes

Decoding

- How can we decode?

1. Without quantization error
   - $y^n = x^n + e^n \Rightarrow s_e = s_y$
   - Decoding algorithms (e.g., the Peterson-Gorenstein-Zierler) for a BCH code, in general, has the following major steps
     1. Detection (to determine the *number* of errors)
     2. Localization (to find the *location* of errors)
     3. Calculation (to calculate the *magnitude* of errors)
2. With quantization error
   - $y^n = x^n + q^n + e^n \Rightarrow s_y = s_e + s_q$
   - Modify the above algorithm
   - Each step becomes an estimation problem
   - Least squares solution largely improves the decoding accuracy

The PGZ algorithm

Detection without quantization

1. Detection ($\nu =?$)

$$\mathbf{S}_t = \begin{bmatrix} s_1 & s_2 & \dots & s_t \\ s_2 & s_3 & \dots & s_{t+1} \\ \vdots & \vdots & \ddots & \vdots \\ s_t & s_{t+1} & \dots & s_{2t-1} \end{bmatrix}$$

Then, $\nu = \mu$ iff $\mathbf{S}_\nu$ is nonsingular for $\nu = \mu$ but is singular for $\nu > \mu$. This is because

$$\mathbf{S}_\mu = V_\mu D V_\mu^T$$

$$V_\mu = \begin{bmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ X_1^{\mu-1} & \dots & X_\mu^{\mu-1} \end{bmatrix}, D = \begin{bmatrix} Y_1X_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & Y_\mu X_\mu \end{bmatrix}$$

The PGZ Algorithm

Detection with quantization

Assume there are $\nu \leq t$ errors. Form $\tilde{\mathbf{S}}_t$

$$\tilde{\mathbf{S}}_t = \begin{bmatrix} \tilde{s}_1 & \tilde{s}_2 & \dots & \tilde{s}_t \\ \tilde{s}_2 & \tilde{s}_3 & \dots & \tilde{s}_{t+1} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{s}_t & \tilde{s}_{t+1} & \dots & \tilde{s}_{2t-1} \end{bmatrix}$$

Existing Approach

1. Set an empirical threshold γ
2. If $\prod \text{eig}(\tilde{\mathbf{S}}_t^H \tilde{\mathbf{S}}_t) < \gamma^2$ then remove the last row and column to find $\tilde{\mathbf{S}}_{t-1}$
3. Continue step 2 until $\prod \text{eig}(\tilde{\mathbf{S}}_\mu^H \tilde{\mathbf{S}}_\mu) \geq \gamma^2$, then $\nu = \mu$

Equivalently we can start from $\tilde{\mathbf{S}}_1$ and go up to $\tilde{\mathbf{S}}_{\mu+1}$.

The PGZ Algorithm

Error Detection

Proposed Approach

Form $\tilde{\mathbf{L}}_{t,t}$ where

$$\tilde{\mathbf{L}}_{\nu,t} = \begin{bmatrix} \tilde{s}_1 & \tilde{s}_2 & \dots & \tilde{s}_\nu \\ \tilde{s}_2 & \tilde{s}_3 & \dots & \tilde{s}_{\nu+1} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{s}_\nu & \tilde{s}_{\nu+1} & \dots & \tilde{s}_{2\nu-1} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{s}_{2t-\nu} & \tilde{s}_{2t-\nu+1} & \dots & \tilde{s}_{2t-1} \end{bmatrix}$$

1. Set an empirical threshold γ'
2. If $\prod \mathrm{eig}(\tilde{\mathbf{L}}_{t,t}^H \tilde{\mathbf{L}}_{t,t}) < \gamma'^2$ then remove the last row and column to find $\tilde{\mathbf{L}}_{t-1,t}$
3. Continue step 2 until $\prod \mathrm{eig}(\tilde{\mathbf{L}}_{\mu,t}^H \tilde{\mathbf{L}}_{\mu,t}) \geq \gamma'^2$, then $\nu = \mu$

Equivalently we can start from $\tilde{\mathbf{S}}_1$ and go up to $\tilde{\mathbf{S}}_{\mu+1}$.

The PGZ Algorithm

Comparison

Consider the extreme case where $\nu = 1$ then

Existing approach:

The decision is based on one sample, i.e., $\tilde{s}_1$

$$\tilde{\mathbf{S}}_1 = \tilde{s}_1 \qquad \Rightarrow \qquad \mathrm{eig}(\tilde{\mathbf{S}}_1^H \tilde{\mathbf{S}}_1) = |\tilde{s}_1|^2 \quad \gtrless_{\nu=0}^{\nu\geq 1} \quad \gamma_1^2$$

The PGZ Algorithm
Comparison

Consider the extreme case where $\nu = 1$ then

Existing approach:

The decision is based on one sample, i.e., $\tilde{s}_1$

$$\tilde{\mathbf{S}}_1 = \tilde{s}_1 \qquad \Rightarrow \qquad \mathrm{eig}(\tilde{\mathbf{S}}_1^H \tilde{\mathbf{S}}_1) = |\tilde{s}_1|^2 \quad \gtrless_{\nu=0}^{\nu\geq 1} \quad \gamma_1^2$$

Proposed approach:

The decision is based on $t-1$ samples, i.e., $\tilde{s}_1$ to $\tilde{s}_{t-1}$

$$\tilde{\mathbf{L}}_{1,t} = \begin{bmatrix} \tilde{s}_1 \\ \tilde{s}_2 \\ \vdots \\ \tilde{s}_{2t-1} \end{bmatrix} \qquad \Rightarrow \qquad \mathrm{eig}(\tilde{\mathbf{L}}_{1,t}^H \tilde{\mathbf{L}}_{1,t}) = \sum_{i=1}^{2t-1} |\tilde{s}_i|^2 \quad \underset{\nu=0}{\overset{\nu\geq 1}{\gtrless}} \quad (2t-1)\gamma_1^2$$

The PGZ Algorithm
Comparison

Consider the extreme case where $\nu = 1$ then

Existing approach:

The decision is based on one sample, i.e., $\tilde{s}_1$

$$\tilde{\mathbf{S}}_1 = \tilde{s}_1 \qquad \Rightarrow \qquad \mathrm{eig}(\tilde{\mathbf{S}}_1^H \tilde{\mathbf{S}}_1) = |\tilde{s}_1|^2 \quad \gtrless_{\nu=0}^{\nu\geq 1} \quad \gamma_1^2$$

Proposed approach:

The decision is based on $t-1$ samples, i.e., $\tilde{s}_1$ to $\tilde{s}_{t-1}$

$$\tilde{\mathbf{L}}_{1,t} = \begin{bmatrix} \tilde{s}_1 \\ \tilde{s}_2 \\ \vdots \\ \tilde{s}_{2t-1} \end{bmatrix} \qquad \Rightarrow \qquad \mathrm{eig}(\tilde{\mathbf{L}}_{1,t}^H \tilde{\mathbf{L}}_{1,t}) = \sum_{i=1}^{2t-1} |\tilde{s}_i|^2 \quad \underset{\nu=0}{\overset{\nu\geq 1}{\gtrless}} \quad (2t-1)\gamma_1^2$$

New decision rule is more reliable than the existing one as it is based on several samples.

The PGZ Algorithm
Error Localization

Error-locator polynomial is defined as

$$\Lambda(x) = \prod_{i=1}^{\nu}(1 - xX_i) = \Lambda_0 + \Lambda_1 x + \ldots + \Lambda_\nu x^\nu$$

- The roots of $\Lambda(x)$, i.e. $X_1^{-1}, \ldots, X_\nu^{-1}$, give the reciprocals of of error locators.
- The coefficients of $\Lambda(x)$, are found by solving end

$$s_j\Lambda_\nu + s_{j+1}\Lambda_{\nu-1} + \cdots + s_{j+\nu-1}\Lambda_1 = -s_{j+\nu},$$

for $j = 1, \ldots, 2t - \nu$, $\nu \leq t$.

The PGZ Algorithm
Error Localization

To find $[\Lambda_1, \ldots, \Lambda_\nu]^T$ we can solve

$$\underbrace{\begin{bmatrix} \tilde{s}_1 & \tilde{s}_2 & \ldots & \tilde{s}_\nu \\ \tilde{s}_2 & \tilde{s}_3 & \ldots & \tilde{s}_{\nu+1} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{s}_\nu & \tilde{s}_{\nu+1} & \ldots & \tilde{s}_{2\nu-1} \end{bmatrix}}_{\tilde{\mathbf{S}}_\nu} \begin{bmatrix} \Lambda_\nu \\ \Lambda_{\nu-1} \\ \vdots \\ \Lambda_1 \end{bmatrix} = - \begin{bmatrix} \tilde{s}_{\nu+1} \\ \tilde{s}_{\nu+2} \\ \vdots \\ \tilde{s}_{2\nu} \end{bmatrix}. \tag{1}$$

The PGZ Algorithm
Error Localization

To find $[\Lambda_1, \ldots, \Lambda_\nu]^T$ we can solve

$$\underbrace{\begin{bmatrix} \tilde{s}_1 & \tilde{s}_2 & \ldots & \tilde{s}_\nu \\ \tilde{s}_2 & \tilde{s}_3 & \ldots & \tilde{s}_{\nu+1} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{s}_\nu & \tilde{s}_{\nu+1} & \ldots & \tilde{s}_{2\nu-1} \end{bmatrix}}_{\tilde{\mathbf{S}}_\nu} \begin{bmatrix} \Lambda_\nu \\ \Lambda_{\nu-1} \\ \vdots \\ \Lambda_1 \end{bmatrix} = - \begin{bmatrix} \tilde{s}_{\nu+1} \\ \tilde{s}_{\nu+2} \\ \vdots \\ \tilde{s}_{2\nu} \end{bmatrix}. \qquad (1)$$

For $\nu < t$, the result will be more accurate by finding the least squares solution for

$$\underbrace{\begin{bmatrix} \tilde{s}_1 & \tilde{s}_2 & \ldots & \tilde{s}_\nu \\ \tilde{s}_2 & \tilde{s}_3 & \ldots & \tilde{s}_{\nu+1} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{s}_\nu & \tilde{s}_{\nu+1} & \ldots & \tilde{s}_{2\nu-1} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{s}_{2t-\nu} & \tilde{s}_{2t-\nu+1} & \ldots & \tilde{s}_{2t-1} \end{bmatrix}}_{\tilde{\mathbf{L}}_{\nu,t}} \begin{bmatrix} \Lambda_\nu \\ \Lambda_{\nu-1} \\ \vdots \\ \Lambda_1 \end{bmatrix} = - \begin{bmatrix} \tilde{s}_{\nu+1} \\ \tilde{s}_{\nu+2} \\ \vdots \\ \tilde{s}_{2\nu} \\ \vdots \\ \tilde{s}_{2t} \end{bmatrix}. \qquad (2)$$

The PGZ Algorithm
Error Localization

LS for error localization (step 2)

- The accuracy of the LS estimation depends on the number of equations per unknowns which is $\frac{2t-\nu}{\nu}$
- It improves when the number of errors (unknowns) decreases

The PGZ Algorithm

Error Localization

LS for error localization (step 2)

- The accuracy of the LS estimation depends on the number of equations per unknowns which is $\frac{2t-\nu}{\nu}$
- It improves when the number of errors (unknowns) decreases

LS for error calculation (step 3)

- The LS is also use to improve the last step of decoding
- The accuracy of estimation, however, depends on the code rate, i.e., $\frac{n-k}{k} = \frac{1}{R} - 1$
- The lower the code-rate, the more accurate the error estimation

Performance Analysis

Linear Reconstruction

Erasure only [Goyal et al, 2001] and [Rath and Guillemot, 2004]

- BCH-DFT codes are tight frames
- The mean squared reconstruction error is minimized by tight frames and is equal to $\mathrm{MSE_q} = \frac{k}{n}\sigma_q^2$

Performance Analysis

Linear Reconstruction

Erasure only [Goyal et al, 2001] and [Rath and Guillemot, 2004]

- BCH-DFT codes are tight frames
- The mean squared reconstruction error is minimized by tight frames and is equal to $\mathrm{MSE}_{\mathrm{q}} = \frac{k}{n}\sigma_q^2$

Erasure and Error

$$\hat{\mathbf{y}} = \mathbf{G}\mathbf{x} + \boldsymbol{\eta}, \qquad \boldsymbol{\eta} = \mathbf{q} + \mathbf{e}$$

$$\hat{\mathbf{x}} = \mathbf{G}^{\dagger}\mathbf{y} = \mathbf{x} + \frac{k}{n}\mathbf{G}^{T}\boldsymbol{\eta}$$

$$\begin{aligned} \mathrm{MSE}_{\mathrm{q+e}} &= \frac{1}{k}\mathbb{E}\{\|\hat{\mathbf{x}} - \mathbf{x}\|^2\} = \frac{1}{k}\mathbb{E}\{\|\frac{k}{n}\mathbf{G}^{T}\boldsymbol{\eta}\|^2\} \\ &= \frac{k}{n}\left[\sigma_q^2 + \frac{\nu}{n}\sigma_e^2\right], \end{aligned} \quad (3)$$

Performance Analysis

Linear Reconstruction

Using BCH-DFT codes, without error correction but merely using linear reconstruction, $\mathrm{MSE}_{\mathrm{q+e}} \le \sigma_q^2$ is possible

$\mathrm{MSE}_{\mathrm{q+e}} \le \sigma_q^2$ for

$$\frac{\sigma_e^2}{\sigma_q^2} \le \frac{n}{k}\frac{n-k}{\nu} \simeq \frac{n}{k}\frac{2t}{\nu},$$

Performance Analysis

Linear Reconstruction

Using BCH-DFT codes, without error correction but merely using linear reconstruction, $\mathrm{MSE}_{\mathrm{q+e}} \leq \sigma_q^2$ is possible

$\mathrm{MSE}_{\mathrm{q+e}} \leq \sigma_q^2$ for

$$\frac{\sigma_e^2}{\sigma_q^2} \leq \frac{n}{k}\frac{n-k}{\nu} \simeq \frac{n}{k}\frac{2t}{\nu},$$

A the worst case where $\nu = n$, reconstruction error is less than quantization error as long as

$$\sigma_e^2 \leq (\frac{1}{R} - 1)\sigma_q^2.$$

Performance Analysis

MSE for 6-bit quantization

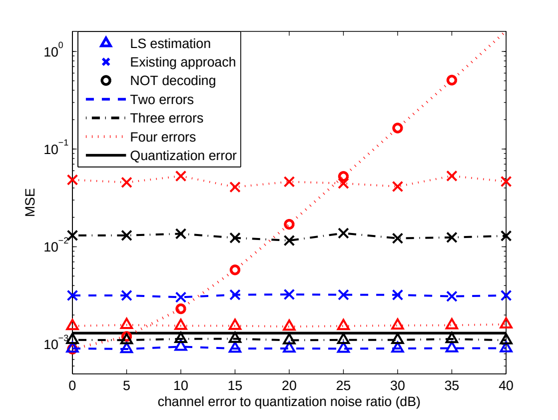

Figure: The LS estimation versus existing approach with perfect error localization for different error patterns in a $(17, 9)$ DFT code.

Performance Analysis

MSE for 6-bit quantization

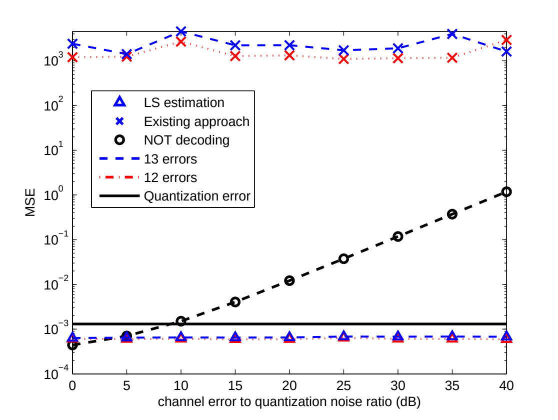

Figure: The MSE performance of a $(36, 9)$ DFT code $(t = 13)$ with perfect error localization.

Performance Analysis

MSE for 6-bit quantization

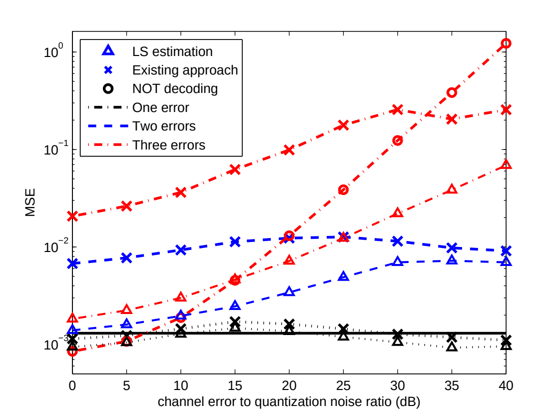

Figure: The LS decoding (detection, localization, and estimation) and existing approach for a $(17, 9)$ DFT code.

Thank you!