“A Robust MMSE Equalizer for MIMO Enhanced CDMA Systems”

Christian Mehlführer

chmehl@nt.tuwien.ac.at

December 7, 2006
Outline

• Introduction and system model
• MMSE equalizer
• Iterative algorithm
• Simulation results
• Summary
In CDMA systems, users are distinguished by orthogonal spreading codes.

Frequency selective channels cause multiple access interference (MAI).

MAI gets worse when:
- number of codes is increased
- number of transmit antennas is increased
- code length is decreased
System Model

\[ N_c = \text{number of spreading sequences} \]
\[ C_{sp} = \text{spreading sequences} \]
\[ C_{sc} = \text{scrambling sequence} \]
MMSE Equalizer

System model:

\[
\mathbf{r}_i = \begin{bmatrix}
\mathbf{r}_i^{(1)} \\
\vdots \\
\mathbf{r}_i^{(NR)}
\end{bmatrix} = \begin{bmatrix}
\mathbf{H}^{(1,1)} & \cdots & \mathbf{H}^{(1,NT)} \\
\vdots & \ddots & \vdots \\
\mathbf{H}^{(NR,1)} & \cdots & \mathbf{H}^{(NR,NT)}
\end{bmatrix} \cdot \begin{bmatrix}
\mathbf{s}_i^{(1)} \\
\vdots \\
\mathbf{s}_i^{(NT)}
\end{bmatrix} + \mathbf{v}_i = \mathbf{Hs}_i + \mathbf{v}_i
\]

Cost function:

\[
J\left(\mathbf{f}^{(n_t)}\right) = E\left\{ |\mathbf{f}^{(n_t)H} \mathbf{r}_i - \mathbf{s}_{i-\tau}^{(n_t)}|^2 \right\}
\]

→ MMSE equalizer:

\[
\mathbf{f}^{(n_t)} = \left[\mathbf{f}^{(1,n_t)T}, \ldots, \mathbf{f}^{(NR,n_t)T}\right]^T = \left(\mathbf{HH}^H + \frac{\sigma_v^2}{\sigma_s^2} \mathbf{I}\right)^{-1} \mathbf{He}_{\tau}^{(n_t)}
\]
MMSE Equalizer

\[ f(n_t) = \left[ f(1,n_t)^T, \ldots, f(N_R,n_t)^T \right]^T = \left( HH^H + \frac{\sigma_v^2}{\sigma_s^2} I \right)^{-1} H e^{(n_t)} \]

- Direct inversion of the matrix requires divisions and high precision → can be a numerical problem in fixed-point implementations

- Idea: calculate matrix inverse iteratively
  - only mult-add operations are required
  - algorithm suitable for fixed-point implementations
Iterative Algorithm

Consider the following iteration:

\[ W_{k+1} = W_k + \mu_k \left( I - W_k A A^H W_k^H \right) W_k \]

\[ A A^H = H H^H + \frac{\sigma_v^2}{\sigma_\delta^2} I \]

Singular Value Decomposition \( W_k A = U \Sigma_k V^H \) leads to:

\[ W_{k+1} A = U \left[ \Sigma_k + \mu_k \left( I - \Sigma_k \Sigma_k^H \right) \Sigma_k \right] V^H \]

\[ \Sigma_{k+1} = \Sigma_k + \mu_k \left( I - \Sigma_k \Sigma_k^H \right) \Sigma_k \]

Main diagonal:

\[ \sigma_{k+1}^{(i)} = \sigma_k^{(i)} + \mu_k \sigma_k^{(i)} \left( 1 - |\sigma_k^{(i)}|^2 \right) \quad i = 1 \ldots N_R L_f \]
Convergence Analysis

Main diagonal:

$$\sigma_{k+1}^{(i)} = \sigma_k^{(i)} + \mu_k \sigma_k^{(i)} \left(1 - |\sigma_k^{(i)}|^2\right) ; i = 1 \ldots N_R L_f$$

The "errors" of the singular values are given by:

$$e_{k+1} = 1 - \sigma_{k+1}^{(i)} = 1 - \sigma_k^{(i)} - \mu_k \sigma_k^{(i)} \left(1 - |\sigma_k^{(i)}|^2\right) =
\underbrace{(1 - \sigma_k^{(i)}) \left(1 - \mu_k \sigma_k^{(i)} \left(1 + \sigma_k^{(i)}\right)\right)}_{e_k}$$

For convergence we require:

$$\frac{|e_{k+1}|}{|e_k|} = \left|1 - \mu_k \sigma_k^{(i)} \left(1 + \sigma_k^{(i)}\right)\right| < 1$$
Step-size Selection

Convergence condition:

\[ 0 < \mu_k < \frac{2}{\sigma_k^{(\text{max})}(1 + \sigma_k^{(\text{max})})} \]

Direct calculation of the maximum singular value requires high computational effort!

→ tight upper bound for the maximum singular value is needed

\[ 0 < \mu_k < \frac{2}{0.25 + 2\|\mathbf{W}_k \mathbf{A} \mathbf{A}^H \mathbf{W}_k^H\|_1} \leq \frac{2}{\sigma_k^{(\text{max})}(1 + \sigma_k^{(\text{max})})} \]
How To Invert A Matrix Using The Iterative Algorithm?

Iterative algorithm:

\[ W_{k+1} = W_k + \mu_k (I - W_k A A^H W_k^H) W_k \]

\[ W_0 = I \]

\[ \mu_k = \frac{1.99}{0.25 + 2\|W_k A A^H W_k^H\|_1} \]

Algorithm converges to:

\[ \lim_{k \to \infty} W_k A = \lim_{k \to \infty} U \Sigma_k V^H = UV^H \]

\[ \lim_{k \to \infty} W_k A A^H W_k^H = UV^H VU^H = I \]

\[ \lim_{k \to \infty} W_k^H W_k = (A A^H)^{-1} \]

\[ AA^H = HH^H + \frac{\sigma_v^2}{\sigma_s^2} I \]
Special Cases

• No Iteration, starting value \( W_0 = I \)

\[
W_0^H W_0 = I \quad \rightarrow \quad f^{(n_t)} = H e^{(n_t)}_T
\]

\( \rightarrow \) matched filter

• 1 Iteration, starting value \( W_0 = I \)

\[
W_1 = (1 + \mu_0) I - \mu_0 \left( HH^H + \frac{\sigma_v^2}{\sigma_s^2} I \right)
\]

\[
f^{(n_t)} = W_1^H W_1 H e^{(n_t)}_T
\]

\( \rightarrow \) 2 matrix-vector multiplications
General Case

\[ W_{k+1} = W_k \left[ (1 + \mu_k) I - \mu_k \left( HH^H + \frac{\sigma_v^2}{\sigma_s^2} I \right) W_k^H W_k \right] \]

\[ f^{(n_t)} = W_{k+1}^H W_{k+1} \text{He}_{\tau}^{(n_t)} \]

→ complexity roughly 2.5 matrix-matrix multiplications per iteration

once \( W \) is known, the equalizers for all streams are known
Simulation parameters:

- Code rate: 0.2
- 10 codes of length 16
- Pedestrian B channel model
- Block fading
- $L_h = 15$ chips
- $L_f = 5L_h$
- Perfect channel knowledge
- Perfect noise knowledge
2x2 High Channel Coding Rate

Simulation parameters:
- Code rate: 0.7
- 10 codes of length 16
- Pedestrian B channel model
- Block fading
- $L_h = 15$ chips
- $L_f = 5L_h$
- Perfect channel knowledge
- Perfect noise knowledge

Simulation parameters:
- Code rate: 0.7
- 10 codes of length 16
- Pedestrian B channel model
- Block fading
- $L_h = 15$ chips
- $L_f = 5L_h$
- Perfect channel knowledge
- Perfect noise knowledge
## Simulation Results

### SNR loss in dB at BLER=0.1

<table>
<thead>
<tr>
<th>Iteration No.</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1x1, R=0.2</td>
<td>4.0</td>
<td>0.9</td>
<td>0.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1x1, R=0.7</td>
<td>N/A</td>
<td>2.3</td>
<td>1.0</td>
<td>0.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2x2, R=0.2</td>
<td>6.0</td>
<td>3.3</td>
<td>1.7</td>
<td>0.6</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2x2, R=0.7</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>2.1</td>
<td>0.9</td>
<td>0.3</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4x4, R=0.2</td>
<td>5.9</td>
<td>4.1</td>
<td>2.7</td>
<td>1.6</td>
<td>0.7</td>
<td>0.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4x4, R=0.7</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>2.6</td>
<td>1.5</td>
<td>0.8</td>
<td>0.3</td>
</tr>
</tbody>
</table>
Summary

• Robust iterative algorithm for matrix inversion was presented
• Algorithm requires only mult-add operations
• Number of required iterations increases with order of the MIMO system and the code rate
• Complexity reductions for time-variant channels are expected
Thank you for your attention!