7.2 Linear Interference Cancellation

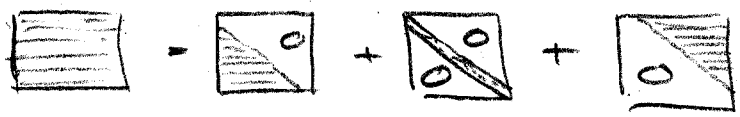
- Linear interference cancellation refers to iterative solution of the ZF or MMSE equations, in which previous estimates of interferers are subtracted (cancelled) out.
- The converged solution is just a ZF or MMSE receiver. We already know their performance (Section 4.4), so why study them further? Because the form of the equations is instructive, and suggests variations can can be implemented more easily. [Tan00, Rooy00]
- After MF/Rake/MRC reduction to sufficient stats, we want to solve

$$G\underline{b} = \underline{\zeta} \qquad \text{where} \quad G = C^{+}\underline{R}CA \quad \text{ZF}$$

$$G = \left(C^{+}RC + N_0A^{-2}\right)A \quad \text{MMSE}$$

Additive decomposition of G helps:

$$G = L + D + U$$



want $(L+D+U)\underline{b} = \underline{\zeta}$ $\qquad \underline{b} \in \mathcal{E}^{NK}$

How?

- Jacobi iteration (Appendix O) is one way:

$$D\underline{b} = \underline{s} - (L+U)\underline{b} \quad \text{is satisfied by a solution.}$$

The j^{th} iterate is $\underline{b}^{(j)}$, and update as

$$D\underline{b}^{(j)} = \underline{s} - (L+U)\underline{b}^{(j-1)}$$

$$\underline{b}^{(j)} = D^{-1}\left(\underline{s} - (L+U)\underline{b}^{(j-1)}\right)$$

The estimate of bit i (of KN) is determined by previous iteration of all other bits:

$$b_i^{(j)} = D_{ii}^{-1}\left(s_i - \sum_{r\neq i}(L+U)_{i,r}\, b_r^{(j-1)}\right) \qquad i = 1, 2, \cdots, NK$$

Convergence iff all eigenvals of $D^{-1}(L+U)$ are in the unit circle.

- Convergence is usually faster with Gauss-Seidel, in which we recognize that the update is typically performed row by row — so use the updated estimates from previous rows of the <u>same</u> iteration

$$b_i^{(j)} = D_{ii}^{-1}\left(s_i - \sum_{r<i}(L+U)_{i,r}\, b_r^{(j)} - \sum_{r>i}(L+U)_{i,r}\, b_r^{(j-1)}\right)$$

or

$$\underline{b}^{(j)} = D^{-1}\left(\underline{s} - L\,\underline{b}^{(j)} - U\,\underline{b}^{(j-1)}\right)$$

Convergence iff all eigenvals of $(I+D^{-1}L)^{-1}D^{-1}U$ are in the unit circle, since

$$(I + D^{-1}L)\,\underline{b}^{(j)} = -D^{-1}U\,\underline{b}^{(j-1)} + D^{-1}\underline{s}$$

$$\underline{b}^{(j)} = -(I+D^{-1}L)^{-1}D^{-1}U\,\underline{b}^{(j-1)} + (I+D^{-1}L)^{-1}D^{-1}\underline{s}$$

- There are about $(NK)^2$ operations in each iteration sweep, so want the number of iterations to be small. We only have to bring the residual error down to a little less than the noise level, so shouldn't take many.

Block Based Formulation

- Neither of the formulations above recognized or exploited the block structure of the equations.

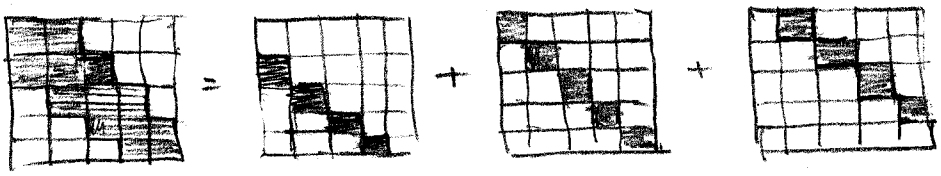
$$G\,\underline{b} = \underline{s}$$

$$\begin{bmatrix} G_{11} & G_{12} & 0 & 0 & 0 & \cdots & 0 \\ G_{21} & G_{22} & G_{23} & 0 & 0 & & 0 \\ 0 & G_{32} & G_{33} & G_{34} & 0 & \cdots & 0 \\ & & \ddots & \ddots & \ddots & & 0 \\ & & & & & & G_{N-1,N} \\ 0 & 0 & 0 & \cdots & & G_{N,N-1} & G_{NN} \end{bmatrix} \begin{bmatrix} \underline{b}(1) \\ \underline{b}(2) \\ \vdots \\ \underline{b}(N) \end{bmatrix} = \begin{bmatrix} \underline{s}(1) \\ \underline{s}(2) \\ \vdots \\ \underline{s}(N) \end{bmatrix}$$

It's organized by K×K blocks. Organize the iteration in the same way.

- Define the block-based additive decomposition of G:

$$G = L_B + D_B + U_B$$



We won't use the tridiagonal structure just yet. For the moment, just express the iteration in block form as follows:

Jacobi

$$D_B \underline{b}^{(j)} = \underline{\xi} - (L_B + U_B)\underline{b}^{(j-1)} \quad \text{as before}$$

For symbol time n,

$$D_{B_{n,n}} \underline{b}^{(j)}(n) = \underline{\xi}(n) - \sum_{r \neq n} \left(L_B + U_B\right)_{n,r} \underline{b}^{(j-1)}(r)$$

Gauss Seidel

$$D_{B_{n,n}} \underline{b}^{(j)}(n) = \underline{\xi}(n) - \sum_{r<n} L_{B_{n,r}} \underline{b}^{(j)}(r) - \sum_{r>n} U_{B_{n,r}} \underline{b}^{(j-1)}(r)$$

Now we have a matrix to invert ($D_{B_{n,n}}^{-1}$) for each time step, so it seems not much point to this approach. However, it can also be solved iteratively (an "inner iteration").

- We now have hybrid approaches

outer iteration	inner iteration
Jacobi	Jacobi
Gauss Seidel	Gauss-Seidel
	other (e.g. conjugate gradient)

The inner iteration can be carried out to effective convergence or left after a few iterations.

- The computation is determined by the number of outer and inner iterations J_{out}, J_{in}. Total computation

$$J_{out}\, N\,(J_{in}+N)\,K^2 \longrightarrow J_{out}\, N\,(J_{in}+2)\,K^2$$

recognizing tridiagonal

- How does performance depend on number of iterations?

Explicit matrix inversion for inner

Allows block size to differ from K

Asynch. Flat AWGN

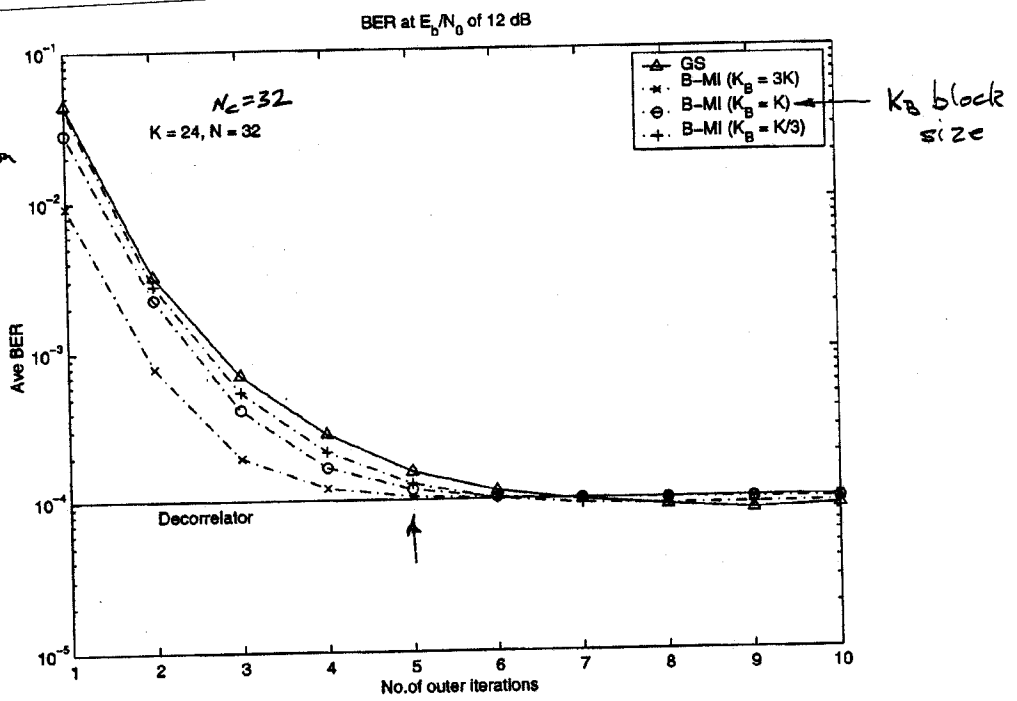
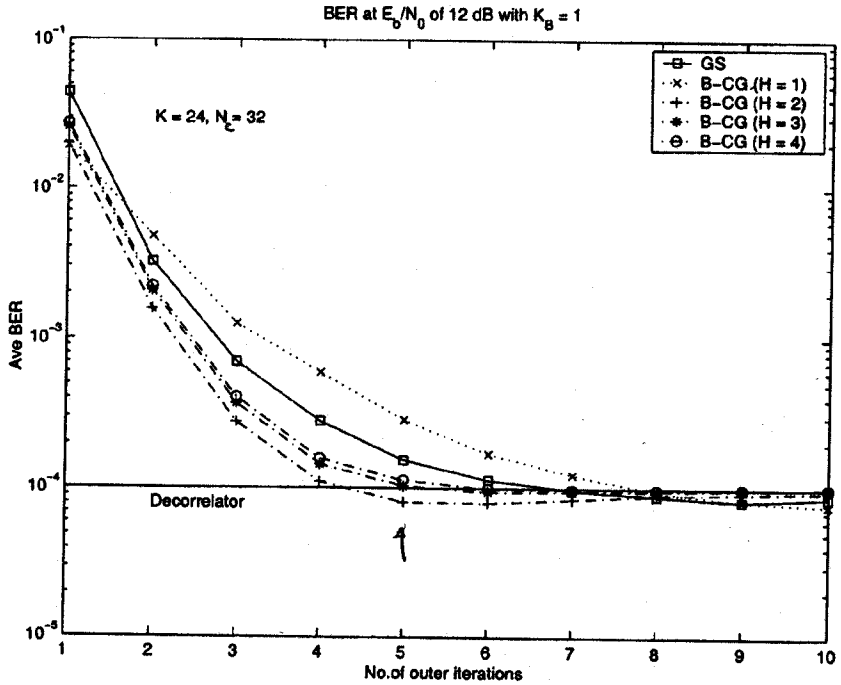


Fig. 5. Averaged BER of the B-MI. Block sizes of $K_B = 8, 24, 72$ are considered where $N = 32$ and $E_b/N_0 = 12$ dB. The point GS is included for reference.

N_c = 32

Conjugate gradient inner iteration.



$H = J_{in}$

CG is best (here) for 2 iterations, but don't trust this value in general.

$J_{in} = 2$

$J_{out} = 5$ is sufficient

Fig. 6. Averaged BER of the B-CG. Block size is $K_B = K$ where $K = 24$, $N = 32$, and $E_b/N_0 = 12$ dB. We consider one, two, three, and four inner iterations and the point GS is included for reference.

- The delay appears to be an inherent N symbol times to accomulate the measurements, plus computation time. Actually, it's much better than that, as shown in the next section

- Note that the approach accommodates long codes.