

Tecniche di Decodifica Iterativa per Codici Concatenati

Marco Ferrari



CNR-IEIIT u.o.s. di Milano



Dipartimento di Elettronica e Informazione
Politecnico di Milano



CNR-IEIIT – Politecnico di Milano



Univ. degli studi di BS

Sommario (1)

- Introduzione - Turbo Codici
 - codici convoluzionali recursivi sistematici
 - concatenazione parallelo
 - decodifica *Soft-In-Soft-Out*: algoritmo BCJR
 - fattore estrinseco: decodifica iterativa
 - I Turbo Codici nella storia della codifica di canale
- Distanza minima e prestazioni asintotiche dei Turbo Codici
 - *error floor*
 - Spettro di Hamming e Union Bound
 - Asintoto per decodificatore a massima verosimiglianza (ML)
 - Interleaver
 - Turbo codici a concatenazione seriale



Sommario (2)

- La decodifica iterativa SISO
 - decodifica MAP dei codici componenti: fattori “estrinseco” e “a priori”
 - analisi della Convergenza tramite Informazione mutua e grafici *EXIT*
 - codici Repeat-Accumulate
- Codici *Low Density Parity Check (LDPC)*
 - definizione e proprietà
 - grafo di Tanner e decodifica iterativa
 - analisi della convergenza su canale BEC
- Codici *LDPC* irregolari
 - definizione e proprietà
 - analisi della convergenza su canale BEC
 - teorema “dell’area”: codici *capacity-achieving*
- Altre applicazioni del “principio turbo” nelle telecomunicazioni (cenni)



Introduzione: i Turbo Codici

Nel maggio del '93 a Ginevra durante l'International Conference on Communication (ICC '93) vengono presentati i Turbo Codici

**NEAR SHANNON LIMIT ERROR - CORRECTING
CODING AND DECODING : TURBO-CODES (1)**

Claude Berrou, Alain Glavieux and Punya Thitimajshima

ovvero uno schema di codifica e decodifica di complessità “implementabile” con prestazioni vicino ai limiti teorici.



CNR-IEIIT – Politecnico di Milano

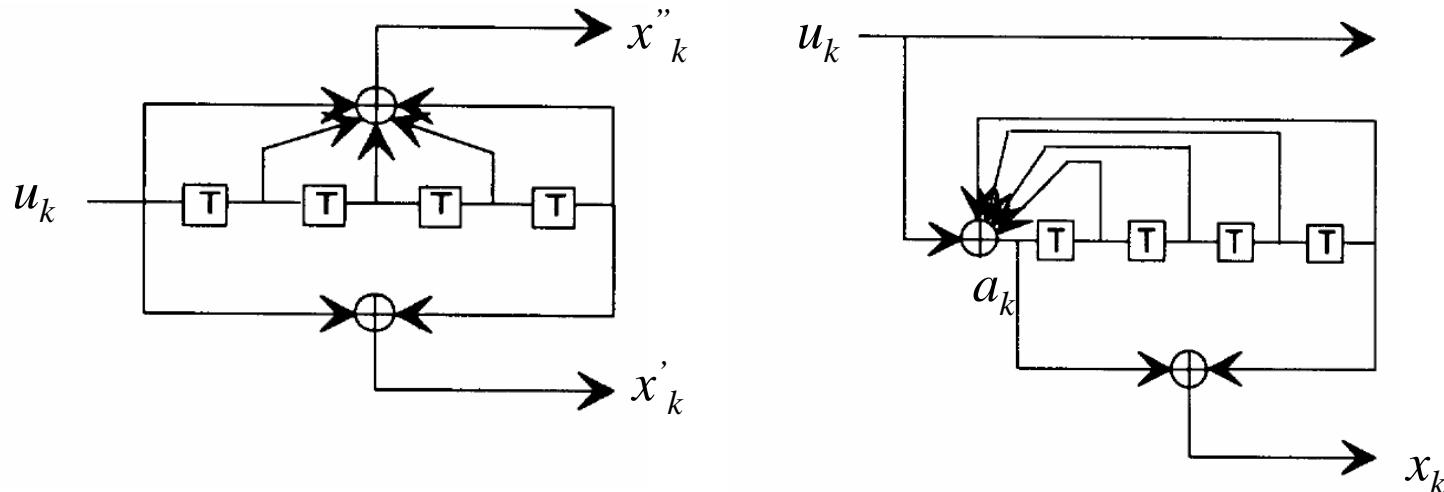


Univ. degli studi di BS

Codici Convoluzionali Recursivi e Sistematici (RSC)

- Codici convoluzionali 16 stati, generatori $g(D)/g_r(D) = 21_{(oct)}/37_{(oct)}$

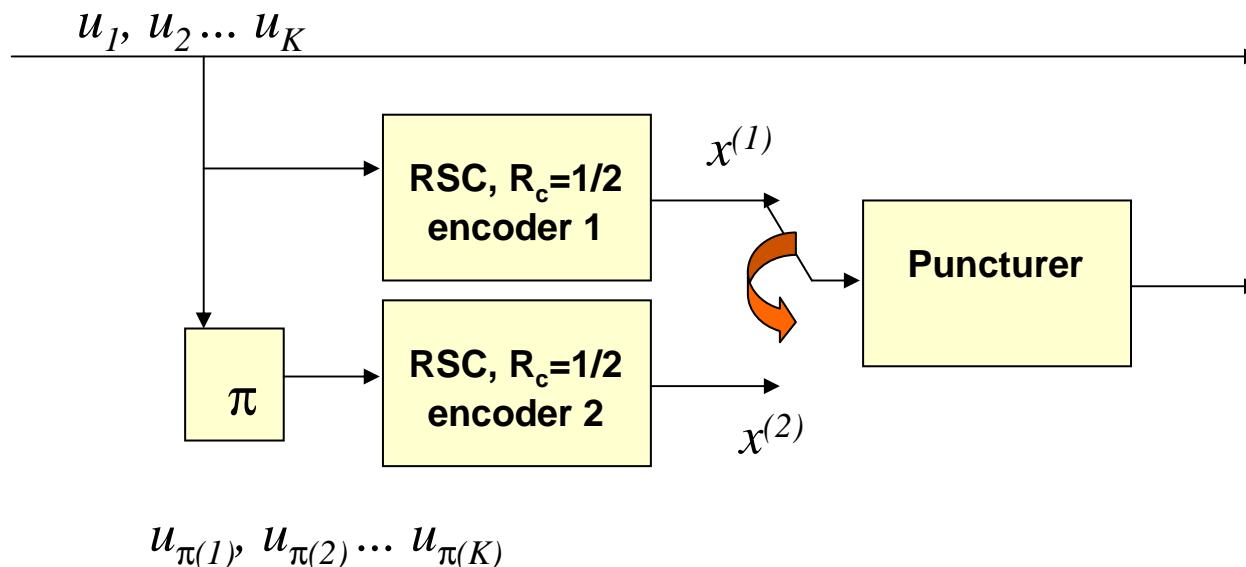
$$a(D) = u(D) + (1 + g_r(D))a(D) \Rightarrow a(D) = \frac{1}{g_r(D)}u(D), \quad x(D) = \frac{g(D)}{g_r(D)}u(D)$$



- si ottiene la sistematicità con lo stesso traliccio del non sistematico con generatori $g(D)$, $g_r(D)$: le sequenze codificate sono le stesse
- se $g(D)u(D)$ non è divisibile per $g_r(D)$ la sequenza $x(D)$ è infinita e periodica di periodo che dipende da $g_r(D)$

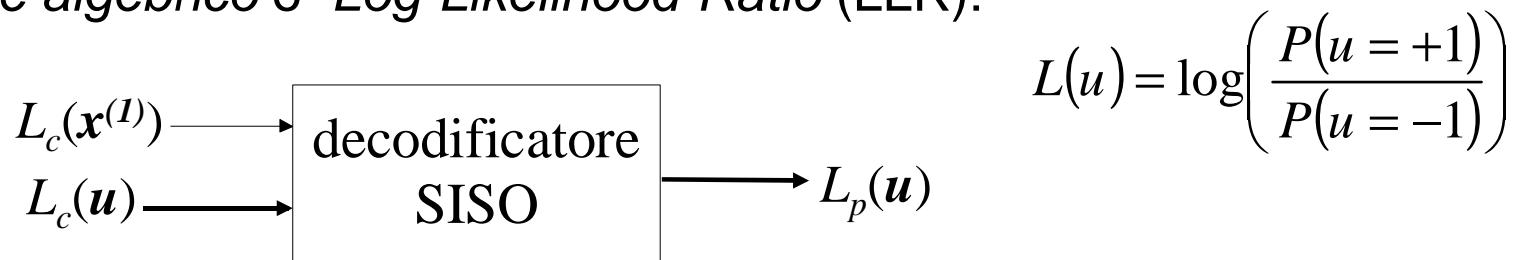
Concatenazione parallelo

- concatenazione di tipo parallelo (PC) tramite interleaver π
- $\pi(i)$ = posizione nella sequenza originale dell'i-simo bit di quella permutata
- legge matematica oppure tabella
- $R=1/3$ con foratura scende a $R=1/2$
- foratura come per i convoluzionali, ma solo sulle parità



Decodificatore SISO (1)

Il decodificatore componente convoluzionale ha ingressi e uscite *soft* (Soft-In-Soft-Out). Per un simbolo binario (+1/-1), basta un numero, per esempio il *valore algebrico* o *Log-Likelihood-Ratio* (LLR):



All'ingresso del decodificatore, da canale AWGN si riceve y invece di u :

$$L_c(u_k) = \log\left(\frac{P(y_k | u_k = +1)}{P(y_k | u_k = -1)}\right) = \frac{2}{\sigma^2} y_k$$

se +1 e -1 equiprobabili a priori

$$= \log\left(\frac{P(u_k = +1 | y_k)}{P(u_k = -1 | y_k)}\right)$$

All'uscita, per il calcolo delle probabilità e del valore algebrico a posteriori:

$$L_p(u_k) = \log\left(\frac{P(u_k = +1 | y_1, y_2, \dots, y_N)}{P(u_k = -1 | y_1, y_2, \dots, y_N)}\right)$$

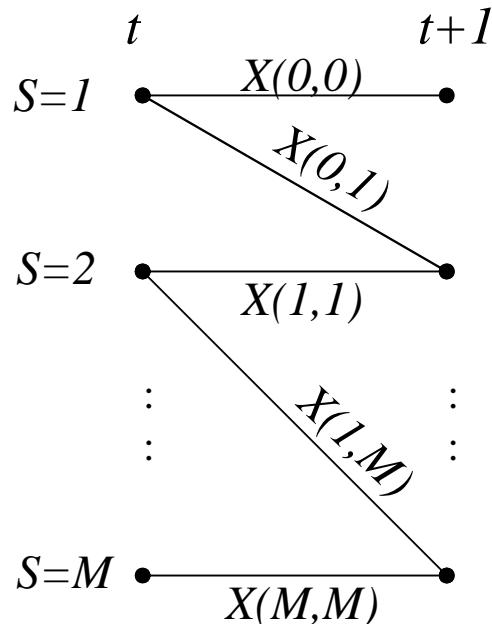
si usa l'algoritmo BCJR, noto anche come MAP (Maximum-a-Posteriori)...



Algoritmo BCJR (1)

Algoritmo proposto nel 1974 da Bahl, Cocke, Jelinek, Raviv, [BCJR74] per il calcolo delle probabilità a posteriori degli stati e delle transizioni di un processo di Markov tempo-discreto, a stati finiti, osservato attraverso un canale rumoroso senza memoria.

Applicazione diretta al traliccio di un codice convoluzionale:



S_t	stato al tempo t
$X(m',m)$	simbolo trasmesso
$u(m',m)$	informazione
Y_1^{τ}	sequenza ricevuta: Y_t è associato alla sezione $(t-1,t)$

[BCJR74] L. Bahl, J. Cocke, F. Jelinek, J. Raviv, “Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate”, IEEE Trans. on Inf. Theory, Mar. 1974, pp. 284-287



Algoritmo BCJR (2)

- Canale senza memoria

$$P[\mathbf{Y}_1^\tau \mid \mathbf{X}_1^\tau] = \prod_{t=1}^{\tau} P[\mathbf{Y}_t \mid \mathbf{X}_t]$$

- Catena di Markov

$$P[\mathbf{Y}_t \mid \mathbf{Y}_{t-1}, S_{t-1} = m] = P[\mathbf{Y}_t \mid S_{t-1} = m]$$

A noi interessa calcolare:

$$P[u_t = b \mid \mathbf{Y}_1^\tau] \propto \sum_{m', m: \mathbf{U}(m', m) = b} \sigma_t(m', m)$$

$$\sigma_t(m', m) = P[S_{t-1} = m', S_t = m, \mathbf{Y}_1^\tau]$$

Il problema diventa quindi il calcolo della probabilità di una certa transizione, congiunto al vettore delle osservazioni; l'algoritmo si basa sulla seguente fattorizzazione:

$$\begin{aligned} \sigma_t(m', m) &= P[\mathbf{Y}_t^\tau, S_t = m \mid S_{t-1} = m'] \cdot P[\mathbf{Y}_1^{t-1}, S_{t-1} = m'] = \\ &= P[\mathbf{Y}_{t+1}^\tau \mid S_t = m] \cdot P[\mathbf{Y}_t, S_t = m \mid S_{t-1} = m'] \cdot P[\mathbf{Y}_1^{t-1}, S_{t-1} = m'] \end{aligned}$$



Algoritmo BCJR (3)

Poniamo:

$$\alpha_{t-1}(m') = P[S_{t-1} = m', \mathbf{Y}_1^{t-1}]$$

$$\gamma_t(m', m) = P[S_t = m, \mathbf{Y}_t | S_{t-1} = m']$$

$$\beta_t(m) = P[\mathbf{Y}_{t+1}^\tau | S_t = m]$$

Esaminando i tre fattori si scopre che gli α si possono calcolare ricorsivamente (*forward recursion*):

$$\begin{aligned}\alpha_t(m) &= \sum_{m'} P[S_t = m, S_{t-1} = m', \mathbf{Y}_t, \mathbf{Y}_1^{t-1}] = \\ \sum_{m'} P[S_t = m, \mathbf{Y}_t | S_{t-1} = m'] P[S_{t-1} = m', \mathbf{Y}_1^{t-1}] &= \sum_{m'} \alpha_{t-1}(m') \gamma_t(m', m)\end{aligned}$$

Con passaggi analoghi per i β si ottiene (*backward recursion*):

$$\beta_t(m) = \sum_{m'} \beta_{t+1}(m') \gamma_{t+1}(m, m')$$



Algoritmo BCJR (4)

I γ dipendono dalla verosimiglianza del ricevuto, e dalla probabilità a priori della transizione (0 oppure $\frac{1}{2}$):

$$\begin{aligned}\gamma_t(m', m) &= P[\mathbf{Y}_t \mid S_t = m, S_{t-1} = m'] P[S_t = m \mid S_{t-1} = m'] = \\ P[\mathbf{Y}_t \mid \mathbf{X}(m', m)] P[S_t = m \mid S_{t-1} = m'] &\underset{AWGN}{\propto} \exp\left(-\frac{|\mathbf{Y}_t - \mathbf{X}|^2}{2\sigma^2}\right) \cdot \frac{1}{2}\end{aligned}$$

Ora non resta che calcolare le probabilità a posteriori delle transizioni:

$$\sigma_t(m', m) = \alpha_{t-1}(m') \gamma_t(m', m) \beta_t(m)$$

Per ottenere i valori algebrici a posteriori dei simboli:

$$L_p(u_t) = \log\left(\frac{P[u_t = +1 \mid \mathbf{Y}_1^\tau]}{P[u_t = -1 \mid \mathbf{Y}_1^\tau]}\right) = \log\left(\frac{\sum_{m', m: u(m', m) = +1} \sigma_t(m', m)}{\sum_{m', m: u(m', m) = -1} \sigma_t(m', m)}\right)$$



Algoritmo BCJR (5)

Riassumendo, volendo calcolare il valore algebrico a posteriori dei bit di un codice convoluzionale, date le \mathbf{Y} osservazioni rumorose dei trasmessi, si può procedere in questo modo:

- Calcolo delle metriche di transizione

$$\gamma_t(m', m) = P[\mathbf{Y}_t \mid \mathbf{X}(m', m)] P[S_t = m \mid S_{t-1} = m']$$

- Forward recursion

$$\alpha_t(m) = \sum_{m'} \alpha_{t-1}(m') \gamma_t(m', m)$$

- Backward Recursion

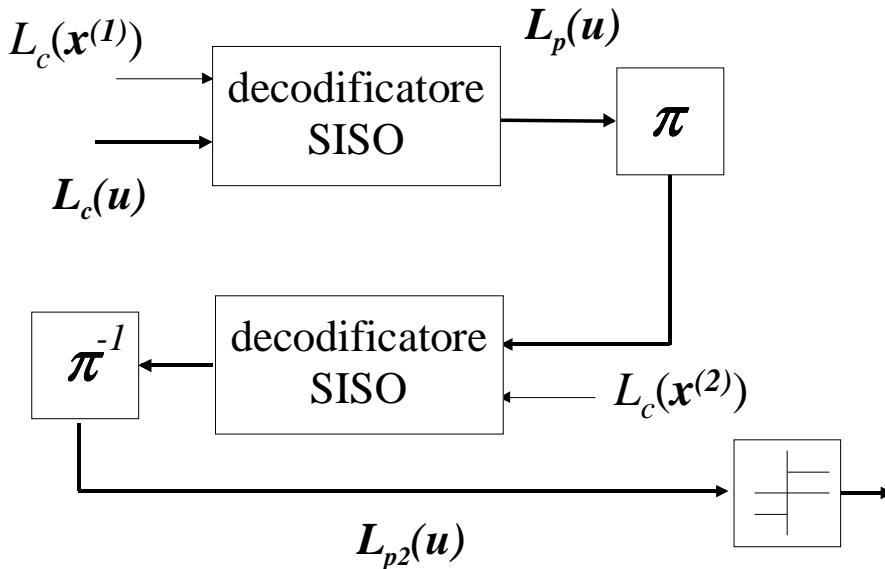
$$\beta_t(m) = \sum_{m'} \beta_{t+1}(m') \gamma_{t+1}(m, m')$$

- Calcolo delle probabilità a posteriori di transizione, e dei valori algebrici di interesse

$$\sigma_t(m', m) = \alpha_{t-1}(m') \gamma_t(m', m) \beta_t(m), \quad L_p(u_t) = \log \left(\frac{\sum_{m': u(m', m)=+1} \sigma_t(m', m)}{\sum_{m': u(m', m)=-1} \sigma_t(m', m)} \right)$$



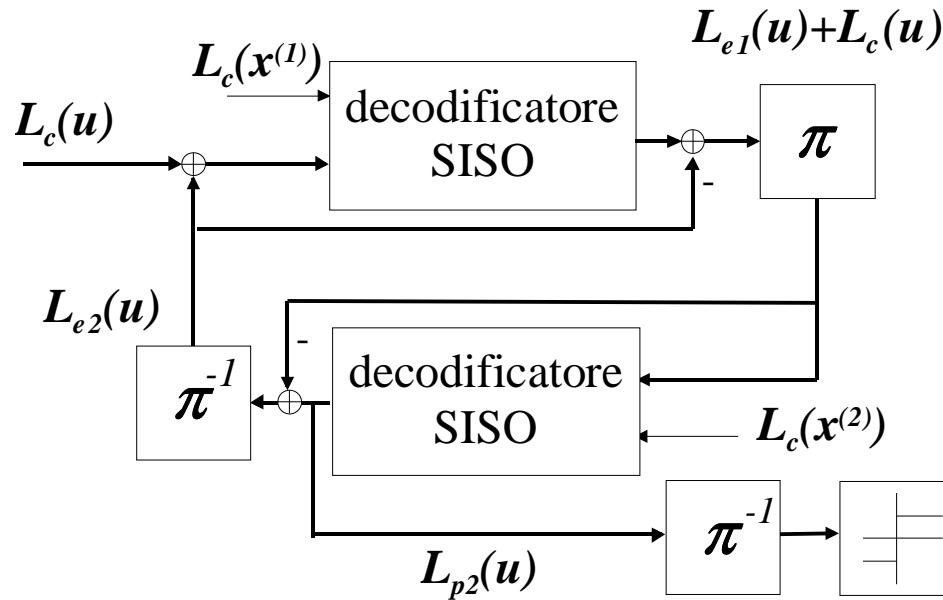
Decodificatore SISO del codice concatenato



- Il primo decodificatore decide solo sulla base delle osservazioni, mentre il secondo si aiuta anche con l'informazione che viene dal primo.
- Se ora ripassassimo questa nuova informazione al secondo, questo potrebbe formulare nuove stime, più affidabili?

Sì, ma a patto di passare informazione “estrinseca”...

Decodificatore SISO iterativo



- Valore algebrico estrinseco perchè il decodificatore la usi come informazione indipendente con profitto: probabilità indipendenti si moltiplicano, i valori algebrici si sommano... In realtà non sono stime indipendenti, ma la presenza dell'interleaver le rende solo debolmente correlate.
- La stima dei bit che esce dal decisore migliora ad ogni iterazione...

Prestazioni dei Turbo Codici presentati a ICC'93

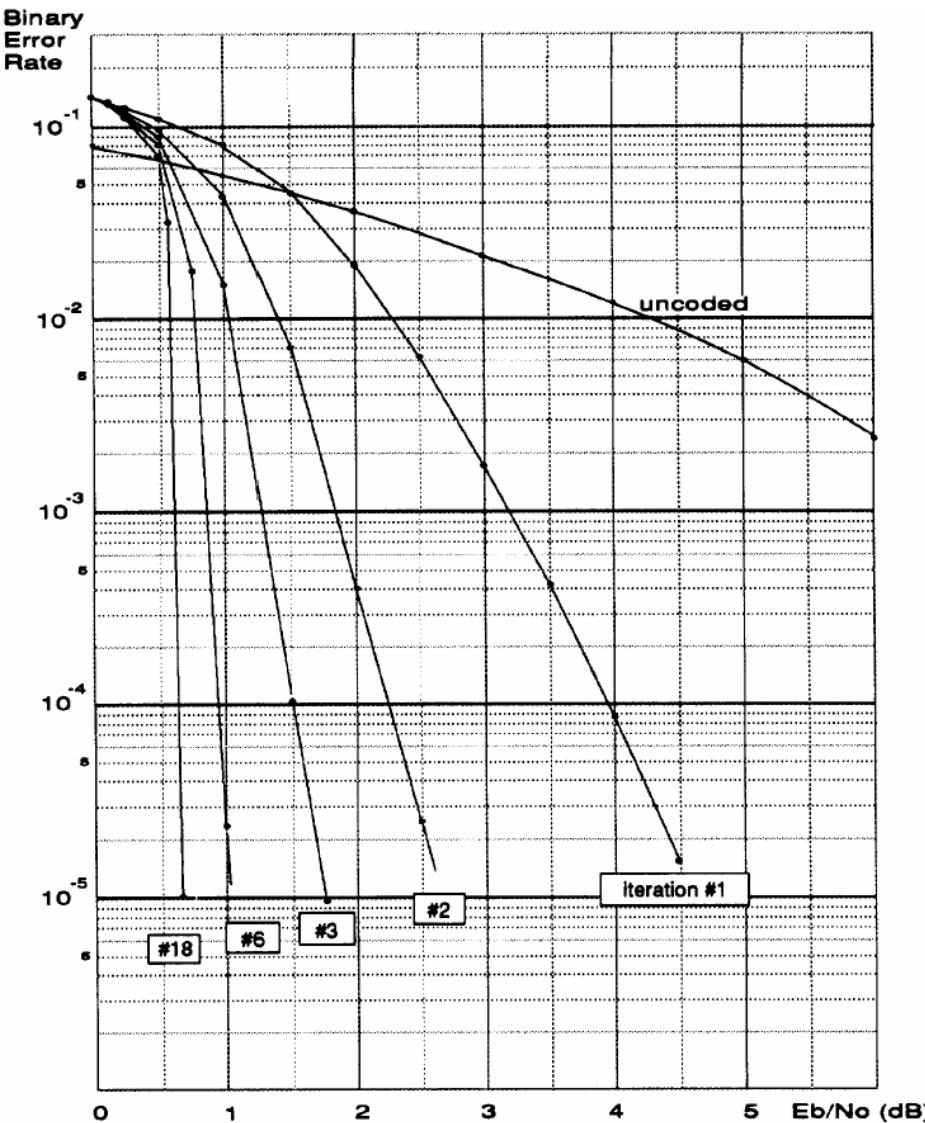
Turbo Codice

codice a blocco

$(N,K)=(131072,65536)$

$R=1/2$,

modulazione QPSK



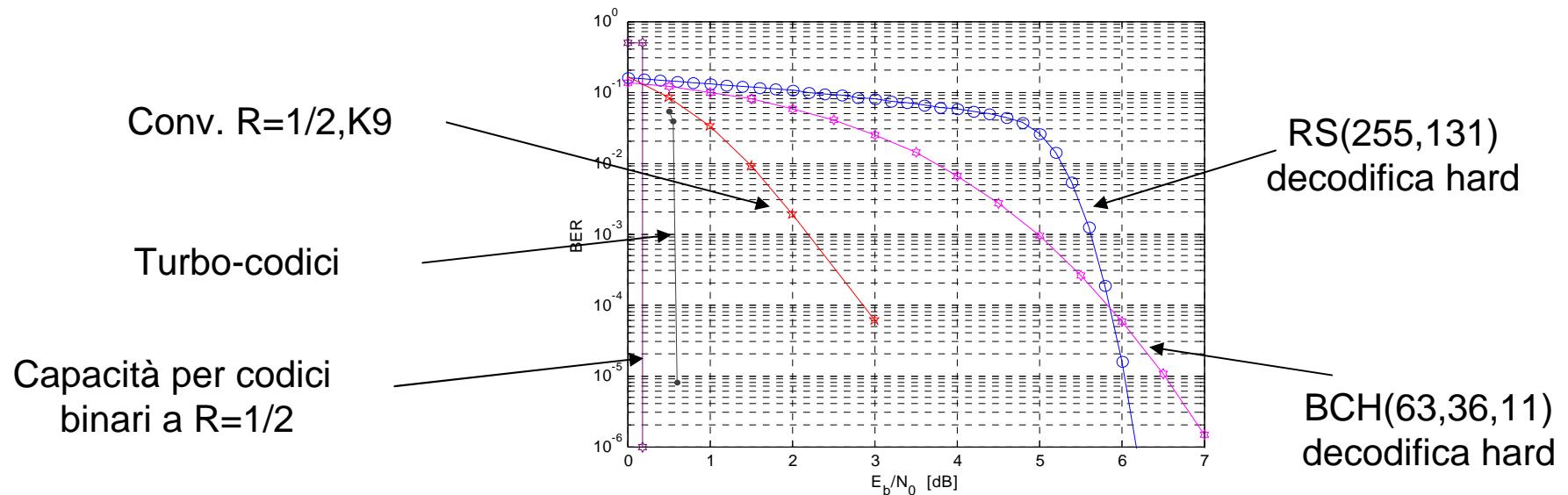
CNR-IEIIT – Politecnico di Milano



Univ. degli studi di BS

“Brevissima” storia della codifica di canale

- Shannon: capacità, esponente d'errore, blocchi grandi, codici random
- Codici a blocco e decodifica *hard*, d_{\min} , codici prodotto, concatenati
- Importante la decodifica a massima verosimiglianza (ML): convoluzionali



I Turbo codici non rientrano in nessuno di questi schemi.

- non sembrano codici costruiti per una grande d_{\min} , ma quant'e'?
- Il blocco è senz'altro grande; la decodifca non è ML, ma cos'e?

Spettro di Hamming

- Il turbo-codice è lineare. La d_{\min} dipende dai codici componenti ma anche dall'interleaver, il che complica le cose. Per calcolare almeno i primi termini dello spettro di Hamming esatto ci vorrà ancora qualche anno [GaPiBe01].
- Un codice binario con spettro d , $n(d)$, $w(d)$, rate R , blocco d'informazione K , peso d'informazione minimo degli $n(d)$ concorrenti pari a w_{\min} con decodifica ML deve comunque sottostare ai lower e upper bound [Rob94]:

$$\frac{w_{\min}}{K} Q\left(\sqrt{\frac{2E_b}{N_0} R d_{\min}}\right) \leq P_b \leq \sum_{d=d_{\min}}^N \frac{w(d)}{K} Q\left(\sqrt{\frac{2E_b}{N_0} R d}\right)$$

- Il turbo-decodificatore non è ML, sembra qualcosa di simile al MAP bit-per-bit che comunque asintoticamente coincide con ML, quindi i bound devono valere anche per la decodifica Turbo anche se non si sa cos'è.

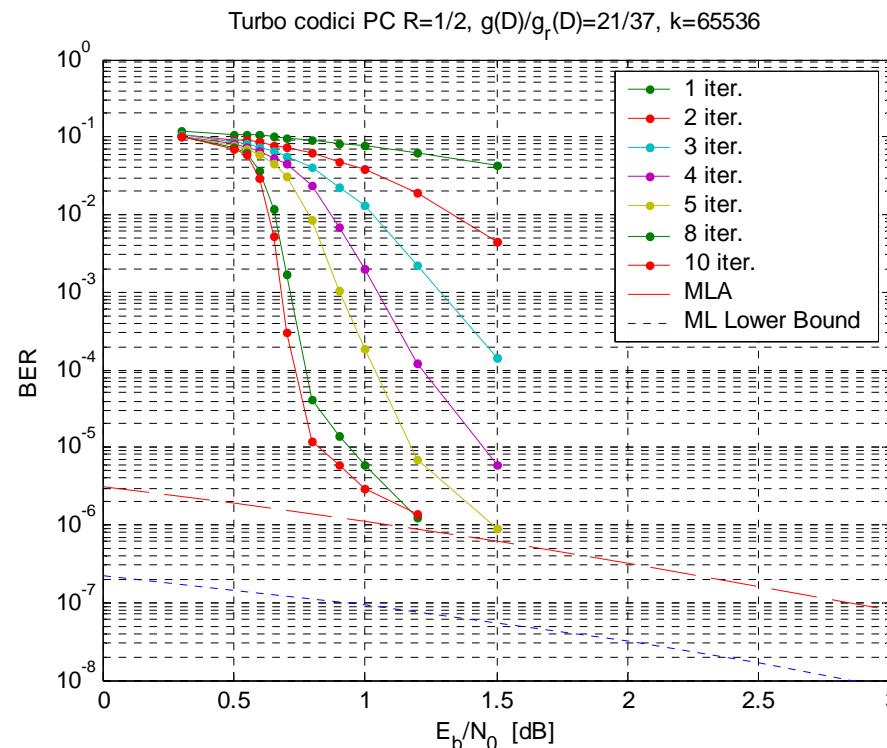
[GaPiBe01] R. Garello, P. Pierleoni, S. Benedetto, “Computing the free distance of turbo codes and serially concatenated codes with interleavers: algorithms and applications”, JSAC vol. 19, No. 5, pp. 800-812, May 2001.

[Rob94] P. Robertson, “Illuminating the Structure of Code and Decoder of Parallel Concatenated Recursive Systematic (Turbo) Codes”, Proc. of Globecom 1994, San Francisco CA, pp. 1298-1303



Error Floor dei Turbo Codici

Alle prime simulazioni di verifica spunta il cosidetto *error floor*: la d_{\min} di un turbo codice simile a quello presentato a ICC è 6, con molteplicità 8, associate a stringhe di informazione di peso di Hamming pari a due.



- Queste osservazioni sono gravide di conseguenze



Prestazioni asintotiche dei Turbo Codici (1)

- Si distinguono due tratti della curva di prestazione: *Waterfall* (basso SNR, rapida riduzione del tasso d'errore con le iterazioni) ed *Error-floor* (alto SNR, bassi tassi d'errore senza ulteriore riduzione con le iterazioni)
- La decodifica iterativa *soft* è “quasi” ML ad alto SNR: prossimità all’MLA asintoto per la decodifica ML dato dall’union bound troncato ai primi termini. A basso SNR non è possibile stimare il decodificatore ML.
- La distanza minima del turbo-codice determina la pendenza dell’error floor, ed insieme alla sua molteplicità, anche il livello di BER a cui questo si assesta.
- La distanza minima (d_{min}) coincide quasi sempre, con interleaver casuale, con il peso minimo di Hamming (d_2) di sequenze di informazione di peso due (grazie all’uso di codici recursivi, d_1 è molto grande).
- Il calcolo certo della distanza minima (d_{min}) è un’operazione proibitiva, a causa della presenza dell’interleaver.



Prestazioni asintotiche dei Turbo Codici (2)

L'*Interleaver uniforme* [BenMon96] ad ogni codifica permuta il blocco di K bit d'informazione con una delle $K!$ permutazioni estratta casualmente: dà una probabilità d'errore sul bit pari al valore atteso di tutte le probabilità d'errore che si ottengono, utilizzando tutti i possibili $K!$ interleaver.

- Upper bound per l'error floor del miglior interleaver possibile (pessimista)
- La distanza minima di un turbo codice pari a $w_{\min} + 2z$ si ottiene quando uno dei $K n^{(c)}_{\min}$ eventi errore del codice componente di pesi $(w_{\min}, w_{\min} + z)$ viene permutato in un'altra di queste $K n^{(c)}_{\min}$ configurazioni dall'interleaver. Sotto interleaving uniforme si ha [BenMon96]:

$$P_b \underset{\substack{Int.Unif. \\ K \rightarrow \infty}}{\sim} n_{\min}^{(c)} w_{\min}! K^{1-w_{\min}}$$

[BenMon96] S. Benedetto, G. Montorsi, "Unveiling Turbo Codes: some results on Parallel Concatenated Coding Schemes", IEEE Trans. on Inf. Theory, Vol. 42, No. 2, Mar. 1996, pp. 409-428



CNR-IEIIT – Politecnico di Milano



Univ. degli studi di BS

Prestazioni asintotiche dei Turbo Codici (3)

$K^{1-w_{min}}$ è detto guadagno d'interleaver: vale 1 con codici non recursivi

- importanza dei codici RSC: d_1 è molto grande, $w_{min}=2$, guadagno K^{-1}
- guadagno d'interleaver: aumentando K l'error floor si abbassa
- Si è quasi subito osservato [Rob94] che un codice componente con polinomio di retroazione primitivo (*) ha un maggior peso di parità z , per eventi errore di peso d'informazione 2 nel codice componente, e quindi dà migliori $d_2 = d_{min}$ nella concatenazione in parallelo. Infatti il periodo coincide con la lunghezza minima degli eventi errore di peso d'informazione 2 nel codice componente.
- Perchè affidarsi solo alla buona sorte? Costruzione più oculata della legge di interleaving...

(*) se $g(D)$ di grado v è primitivo, non è divisore di polinomi del tipo D^q+1 , con $q < 2^v-1$.



Interleaver (1)

- Interleaver deterministici
 - + bassa complessità
 - rischio di grandi molteplicità di parole di peso intermedio; rischio anche con grande peso d'informazione
 - a blocco: buone distanze minime, grandi molteplicità
 - Galois Field prunable (UMTS): evita di permutare un divisore di $g_r(D)$ (polinomio di retroazione) in un altro divisore di $g_r(D)$
 - Relative Prime: $\pi(i) = (s+ip) \bmod K$ con p e K relativamente primi
- Interleaver casuali
 - + buon comportamento con sequenze di informazione di peso grande; distanze minime piccole con sequenze di informazione di peso basso
 - richiede memorizzazione in una tabella



Interleaver (2)

- Interleaver semi-casuali

- S-random - controllo all'estrazione casuale se:

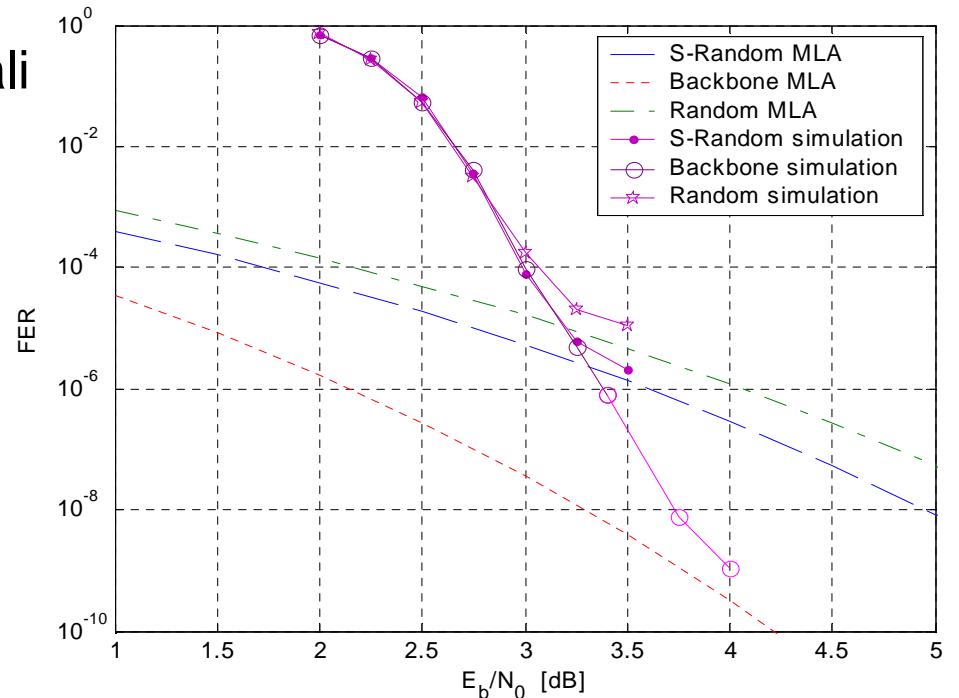
$$|i - j| \leq S \Rightarrow |\pi(i) - \pi(j)| > S, \quad S \cong \sqrt{K/2}$$

- Dithered Relative Prime (DRP):
RP con permutazioni locali casuali
- Backbone [FeBeAmTo03]

Esempio:

Turbo codice $R=3/4, k=1506$

- Random: $d_{min}=6$
- S-Random: $d_{min}=7$
- Backbone: $d_{min}=11$



[FeBeAmTo03] M. Ferrari, S. Bellini, A. Ambrose, M. Tomlinson, "Backbone interleaver design for multi-binary Turbo Codes", Int. Sym. on Turbo Codes, Brest, sept. 2003, pp. 443-446



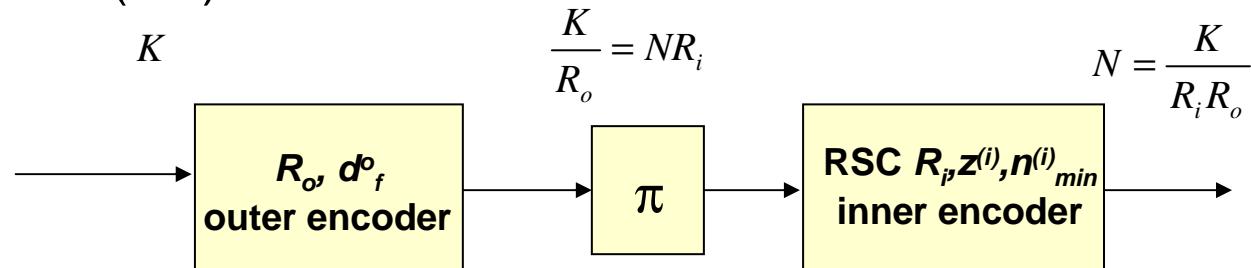
CNR-IEIIT – Politecnico di Milano



Univ. degli studi di BS

Turbo codici Seriali (1)

Un paio d'anni dopo i TC viene proposto uno schema concatenato diverso, di tipo seriale (SC):



$$R = R_i R_o$$

Ricorrendo di nuovo all'uso dell'interleaver uniforme infatti si ottiene che se il codice interno è recursivo, si ha guadagno d'interleaver pari a $d_f^{(o)} / 2$ ($d_f^{(o)}$ pari) [BeDiMoPo98], quindi maggiore che nel caso di codici PC:

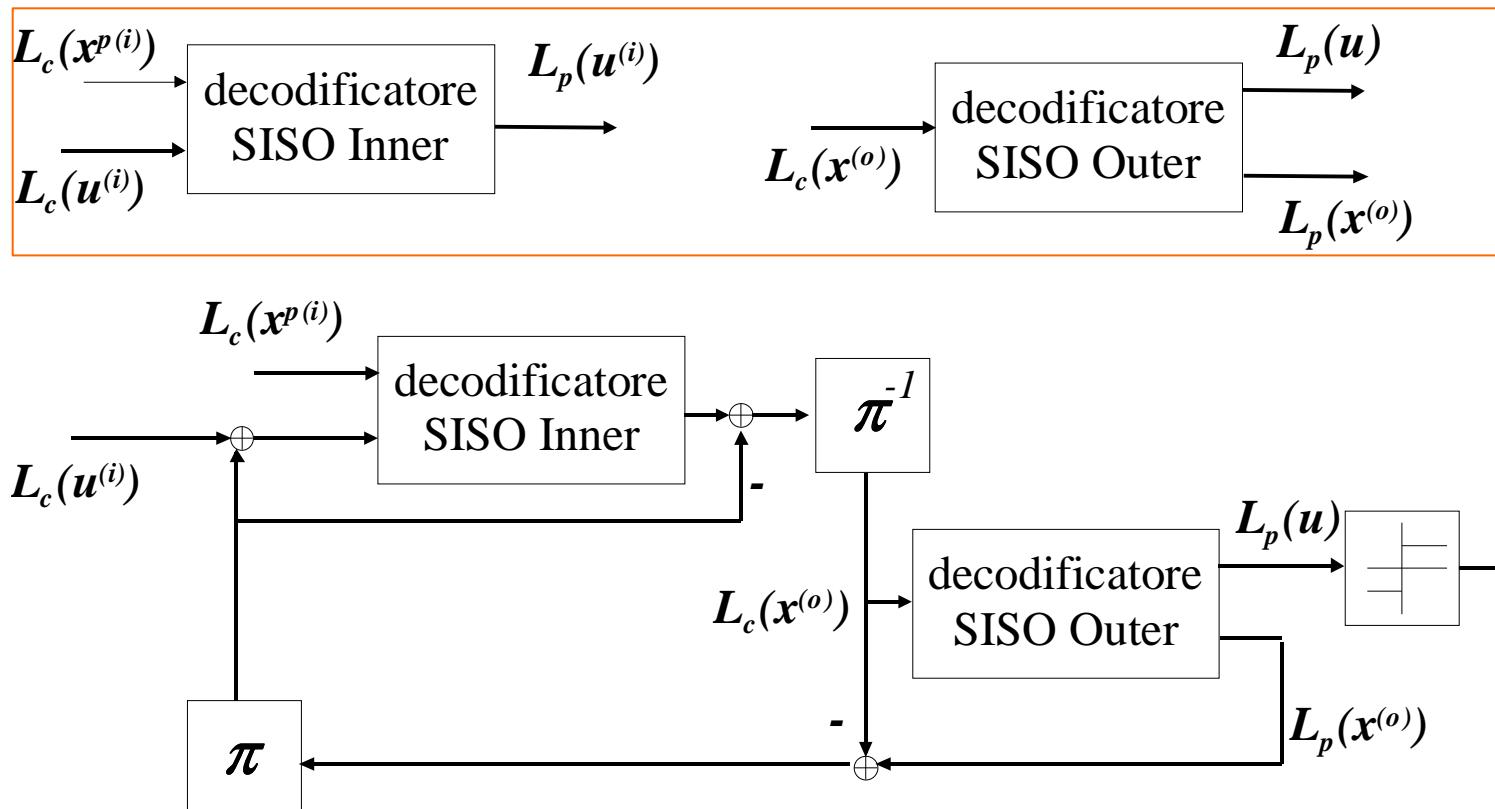
$$P_b \underset{\substack{Int.Unif. \\ N \rightarrow \infty}}{\sim} n_{\min}^{(i)} d_f^{(o)/2} d_f^{(o)}! (NR_i)^{-d_f^{(o)/2}} \xrightarrow[N \rightarrow \infty]{} 0$$

[BeDiMoPo98] S. Benedetto, D. Divsalar, G. Montorsi, F. Pollara, "Serial Concatenation of Interleaved Codes: Performance Analysis, Design, and Iterative Decoding", IEEE Tr. Inf. Theory, Vol. 44, No. 3, May. '98, p. 909-926



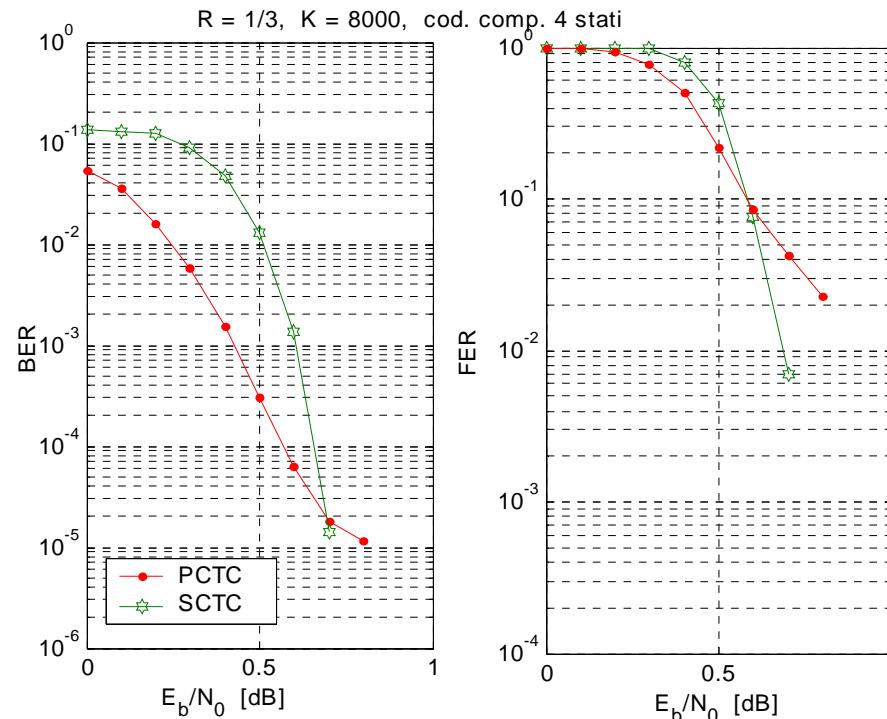
Turbo codici Seriali (2)

Decodifica turbo di un turbo-codice SC: ancora BCJR, ma il decodificatore esterno deve fornire i valori algebrici estrinseci dei bit codificati (per il decodificatore interno), e i valori algebrici a posteriori al decisore:



Turbo codici Seriali (3)

Come previsto dall'interleaver uniforme si ottengono error floor più bassi.



Si trova però anche un peggioramento della “convergenza”...



Decodifica iterativa (1)

Tra i principali “protagonisti” delle ottime prestazioni dei turbo codici c’è lo schema di decodifica iterativa SISO, che abbiamo già visto dare prestazioni prossime al decodificatore ML ad alto SNR. Tuttavia c’è ancora molto da chiarire sulla “decodifica iterativa” che certamente ML non è.

Una rilettura più chiara del ruolo dei fattori estrinseci nell’algoritmo BCJR si trova in [Rob94]. Ripartiamo dal calcolo dei valori algebrici a posteriori:

$$L_p(u_t) = \log \left(\frac{\sum_{m', m: u=+1} \alpha_{t-1}(m') \gamma_t(m', m) \beta_t(m)}{\sum_{m', m: u=-1} \alpha_{t-1}(m') \gamma_t(m', m) \beta_t(m)} \right)$$

$$\begin{aligned} \gamma_t(m', m) &= P[\mathbf{Y}_t | S_t = m, S_{t-1} = m'] P[S_t = m | S_{t-1} = m'] = \\ &P[y_t | u_t(m', m)] P[y_t^{(p)} | x_t^{(p)}(m', m)] P[S_t = m | S_{t-1} = m'] \end{aligned}$$



Decodifica iterativa (2)

Se assumiamo la probabilità a priori di una transizione come probabilità a priori del bit associato, possiamo raccogliere il primo e l'ultimo fattore:

$$L_p(u_t) = \log\left(\frac{P[y_t | u_t = +1]}{P[y_t | u_t = -1]}\right) + \log\left(\frac{P[u_t = +1]}{P[u_t = -1]}\right) + \log\left(\frac{\sum_{m',m:u=+1} \alpha_{t-1}(m') P[y_t^{(p)} | x_t^{(p)}(m',m)] \beta_t(m)}{\sum_{m',m:u=-1} \alpha_{t-1}(m') P[y_t^{(p)} | x_t^{(p)}(m',m)] \beta_t(m)}\right)$$
$$L_p(u_t) = L_c(u_t) + L_a(u_t) + L_e(u_t)$$

- L'LLR estrinseco è dato dalle probabilità del simbolo u_t a posteriori, date tutte le osservazioni degli altri simboli e le loro informazioni a priori, tranne quelle relative al simbolo u_t stesso. E' l'informazione a posteriori che il decodificatore può inferire per u_t sulla base di tutti gli altri simboli e della struttura del codice.
- Nel turbo decodificatore, ogni decodificatore componente usa l'LLR estrinseco dell'altro come se fosse un LLR a priori indipendente.



Convergenza della decodifica Iterativa (1)

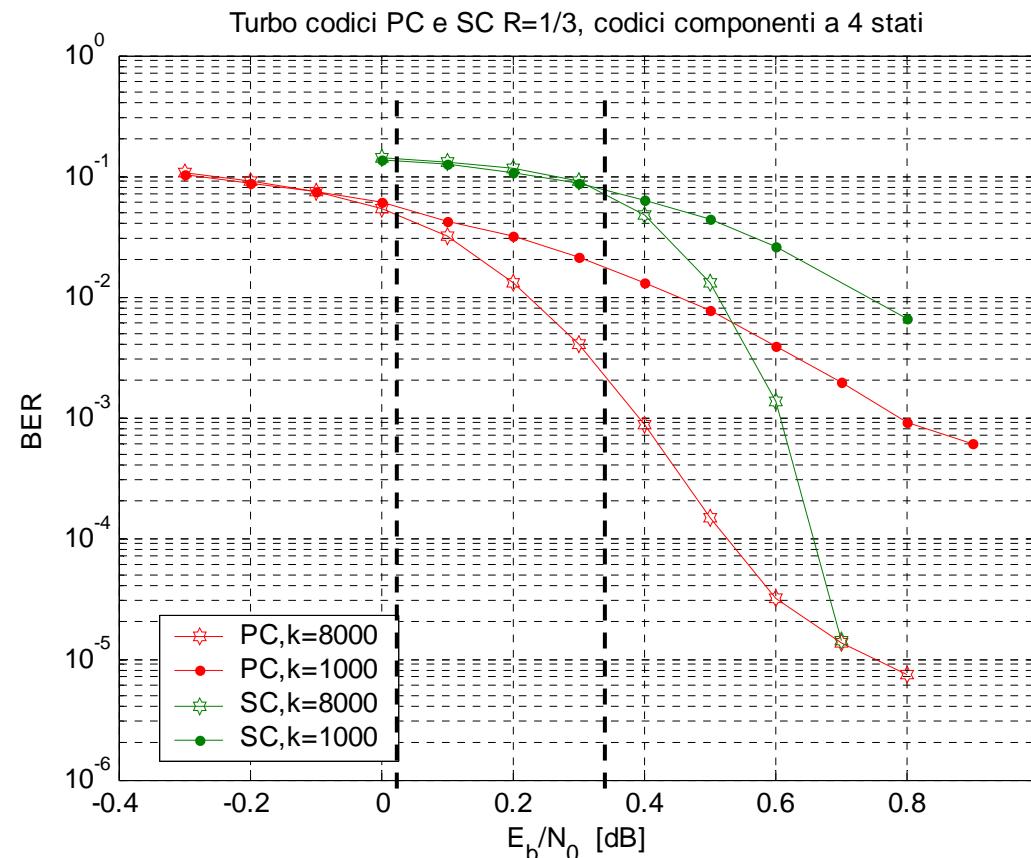
L'interpretazione data per la decodifica turbo può risultare soddisfacente dal punto di vista intuitivo, ma non fornisce strumenti concreti per l'analisi della convergenza dell'algoritmo. Quanto velocemente, con quante iterazioni si trova la soluzione? Sotto quali condizioni e quando invece non si trova? P.e. lo stesso identico discorso si può applicare ai turbo-codici SC, ma perchè lì funziona peggio?

Occorre assumere un punto di vista completamente diverso da quelli tradizionali: la convergenza dell'algoritmo iterativo molto probabilmente non dipende tanto dalle prestazioni che avrebbe il decodificatore ML e cioè dalla forma dello spettro di Hamming. Il problema è quanto buona è la decodifica iterativa di uno schema rispetto ad un altro?



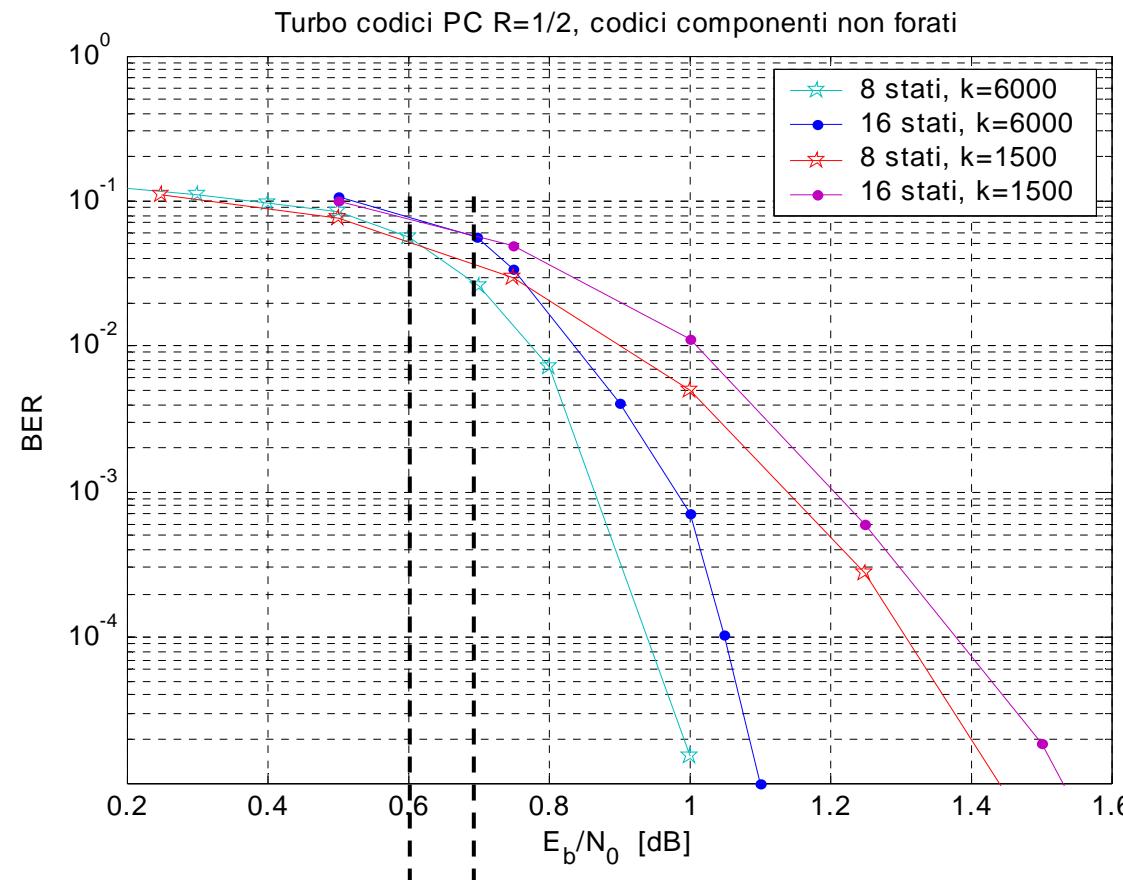
Convergenza della decodifica Iterativa (2)

All'aumentare della lunghezza del blocco le prestazioni migliorano in entrambi gli schemi come previsto dalla teoria oltre un valore “soglia” di E_b/N_0 che è quello che ha senso confrontare con la capacità. Tale soglia è più bassa per i PC che per gli SC...



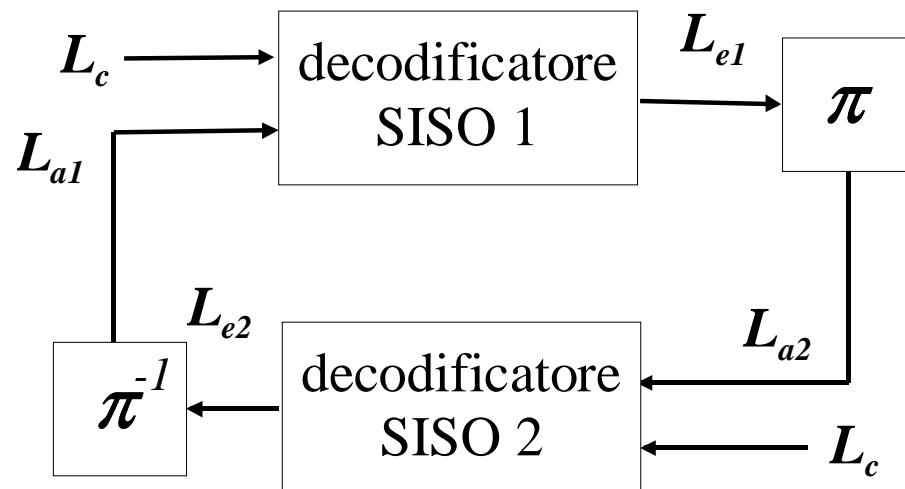
Convergenza della decodifica Iterativa (3)

... più bassa con codici componenti a 8 stati che a 16 stati, meglio con alcuni generatori ($g_r(D)$ non primitivo) che con altri...



Convergenza della decodifica Iterativa (4)

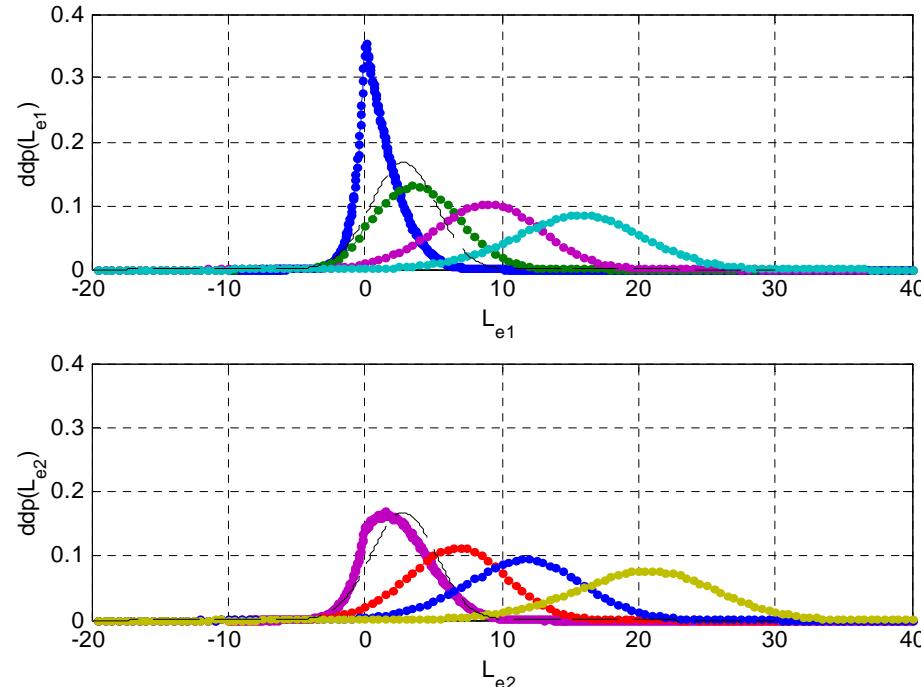
Mettiamoci nell'ottica di avere un blocco di lunghezza infinita: cosa fa la decodifica iterativa di un turbo-codice PC? Un set di L_a con una certa ddp entra in un SISO decoder che genera un set di L_e con un'altra ddp.



Si può studiare la relazione tra queste due ddp. Se il blocco è infinito, si può assumere che ad ogni iterazione gli L_a siano estratti casualmente e indipendentemente con una certa densità di probabilità.

Convergenza della decodifica Iterativa (5)

Esempio di ddp (codice componente Berrou, $E_b/N_0 = 1.5$ dB)



- approssimazione di queste densità con altre densità note: gaussiana
- caratterizzazione della ddp tramite un solo parametro, p.e.

$$L_c = \frac{2}{\sigma^2} y \sim \mathcal{N}\left(\frac{2}{\sigma^2}, \frac{4}{\sigma^2}\right) \Rightarrow L_e \sim \mathcal{N}\left(\frac{2}{\sigma_e^2}, \frac{4}{\sigma_e^2}\right)$$



Informazione mutua (1)

Ad ogni iterazione si ha una diminuzione del BER, perchè passando ad una distribuzione “più a destra”, aumenta l’informazione mutua tra L e X : $p(L/X=+1)$ e $p(L/X=-1)$ si separano, e la probabilità di sbagliare decisione si riduce. Si può usare l’informazione mutua tra L e X per caratterizzare le ddp dei valori algebrici.

Informazione mutua tra una variabile binaria $X = +1/-1$ ed una continua L

$$I(X,L) = H(L) - H(L | X) = H(X) - H(X | L)$$

$$I(X,L) = \sum_x P_X(x) \int p(L | x) \log_2 \left(\frac{p(L | x)}{p(L | +1)P_X(+1) + p(L | -1)P_X(-1)} \right) dL$$

$I(X,L)$ coincide con la capacità di un canale con ingresso discreto binario X e uscita continua L con distribuzione ottima a priori uniforme.

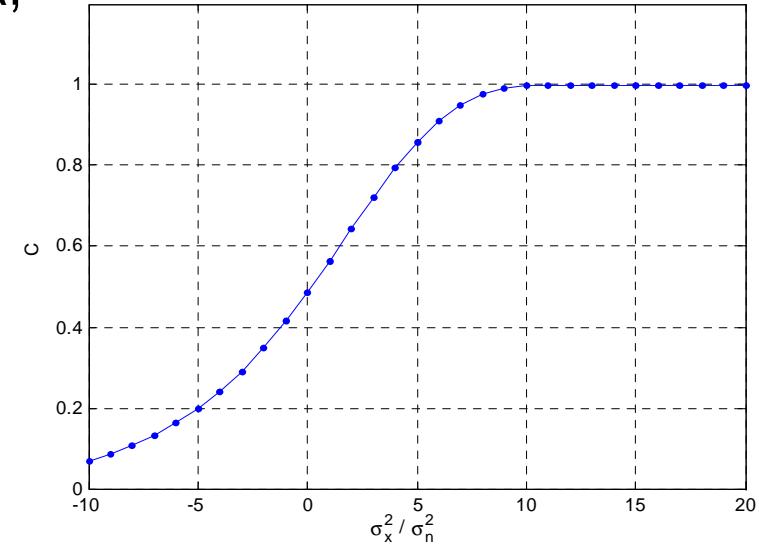


Informazione mutua (2)

- Se la ddp di L fosse davvero gaussiana, $I(X,L)$ avrebbe quindi una forma già nota:

$$L \sim \mathcal{N}(\mu_L, \sigma_L^2) \Rightarrow I(X, L) = C \left(\frac{\sigma_x^2}{\sigma_n^2} = \frac{\mu_L^2}{\sigma_L^2} \right)$$

e la caratterizzazione tramite I sarebbe assolutamente equivalente a quella del SNR.



- Per valutare $I(X,L)$ conviene sfruttare la formulazione inversa:

$$I(X, L) = H(X) - H(X | L) = 1 + E[\log_2(P(x | L))] = 1 + E\left[\log_2\left(\frac{e^{xL/2}}{e^{L/2} + e^{-L/2}}\right)\right]$$

Si può stimare il valor medio con la media campionaria tramite simulazione.

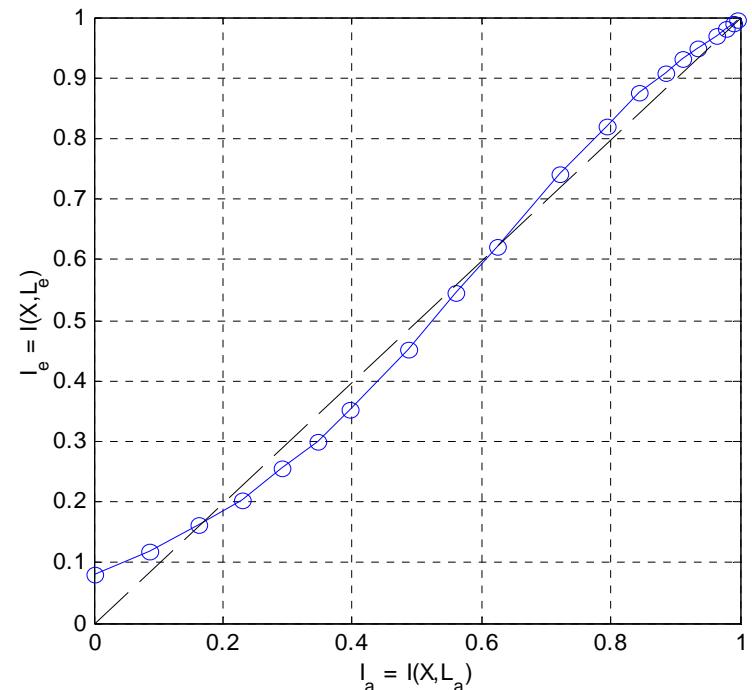


Caratteristica EXIT di un decodificatore SISO

A noi interessa stimare la $I(X, L_e)$ di un decodificatore componente SISO, al variare della $I(X, L_a)$ degli L_a in ingresso, per un certo valore di E_b/N_0 : si ottiene così la curva di trasferimento dell'informazione estrinseca (EXIT [tBr01]) di un codice componente.

Es. $E_b/N_o=0.3 \text{ dB}$, $R=2/3$, (Berrou)

- anche per $I_a=0$, $I_e>0$: la struttura del codice consente di dedurre un po' di informazione su un simbolo basandosi su quella di canale degli altri.
- I_e cresce con I_a , prima più lentamente, poi anche più velocemente.
- per $I_a=1 \Rightarrow I_e=1$ (nessun errore)



[tBr01] S. ten Brink, "Convergence behaviour of iteratively decoded parallel concatenated codes", IEEE Trans. on Comm., vol. 49, No. 10, Oct. 2001, pp. 1727-1737



CNR-IEIIT – Politecnico di Milano



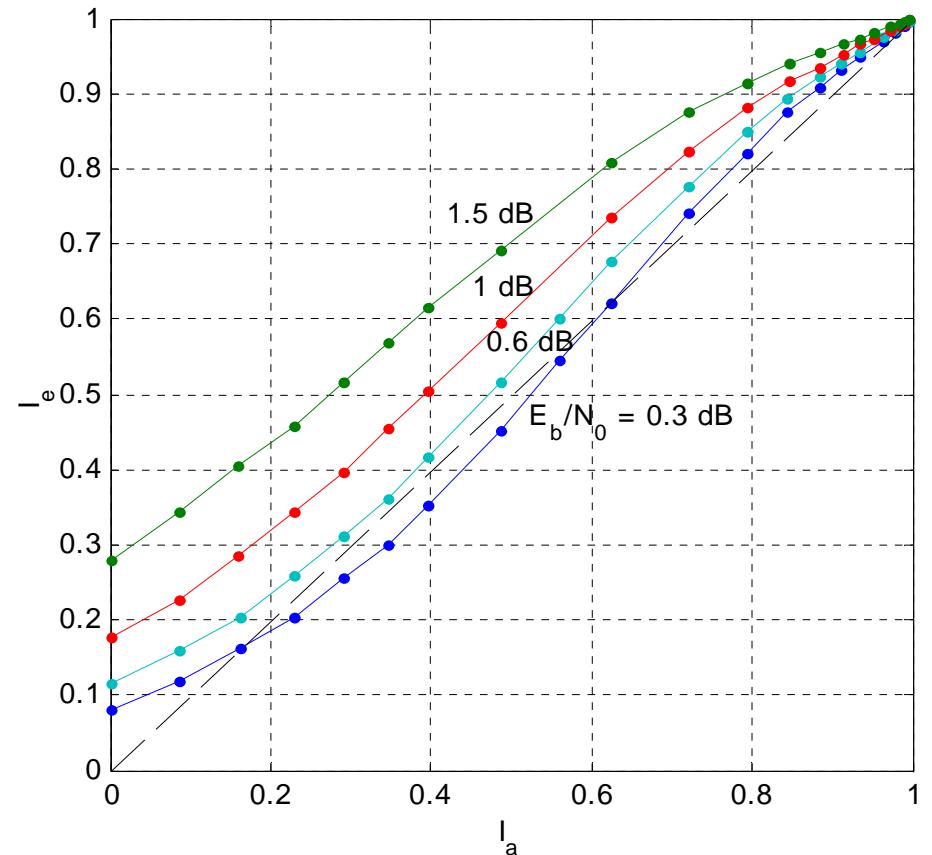
Univ. degli studi di BS

Caratteristica EXIT (2)

Caratteristica EXIT: dipendenza da E_b/N_o

Es. $E_b/N_o = 0.3, 0.6, 1, 1.5 \text{ dB}$, $R=2/3$,
(Berrou)

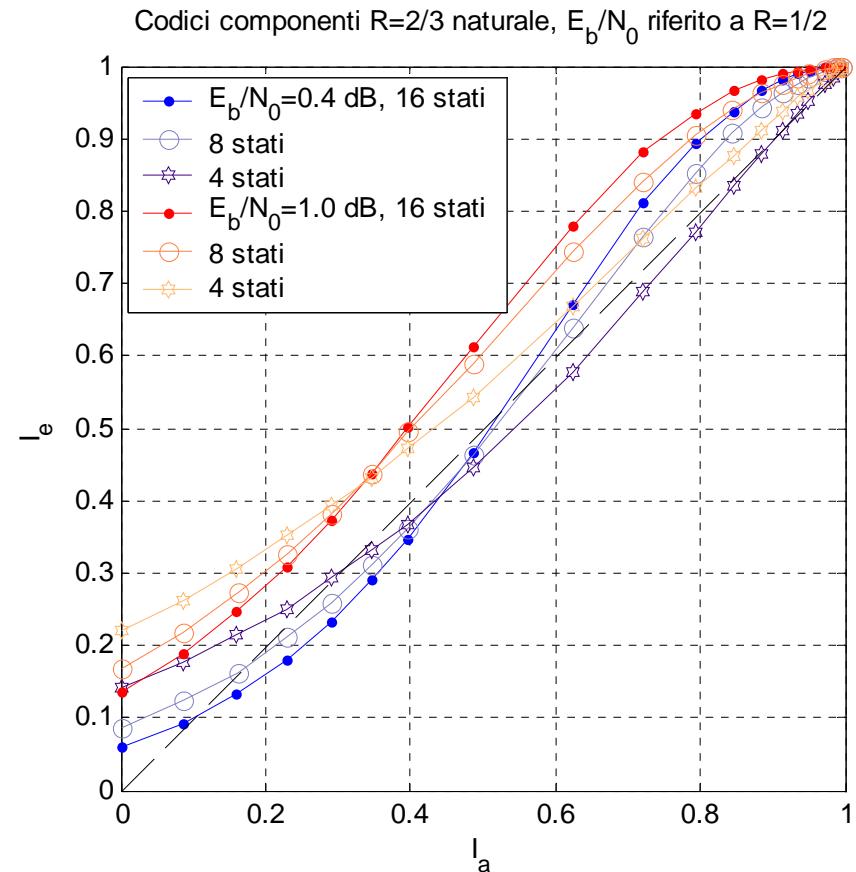
- per $I_a=0$, I_e aumenta con E_b/N_o l'informazione di canale dei simboli su cui si basa il decodificatore aumenta.
- si riduce il range in cui I_e cresce più lentamente di I_a , fino a scomparire.
- I_e tende a uno più rapidamente.



Caratteristica EXIT (3)

Curve EXIT di codici componenti di varie memorie

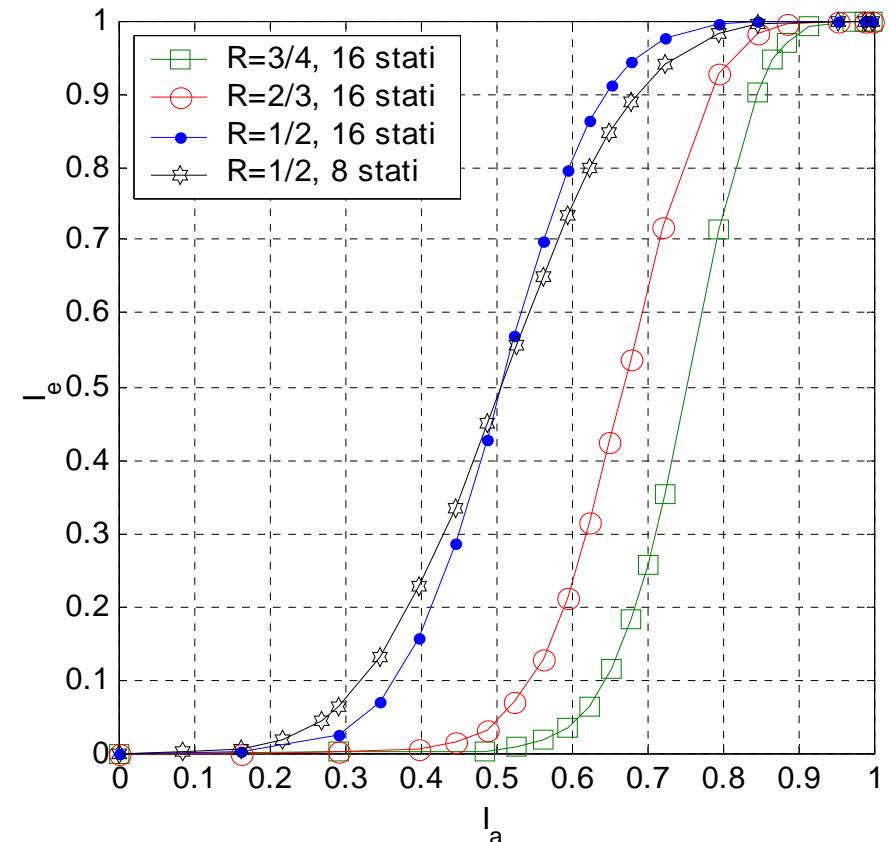
- la memoria del codice più alta è controproducente a basse I_a , poi fa sì che I_e aumenti più rapidamente: codici più potenti danno prestazioni peggiori di codici meno potenti sotto capacità.
- comportamento simile al variare di E_b/N_o , ma il punto di intersezione si sposta verso sinistra, fino a scomparire.



Caratteristica EXIT (4)

Curve EXIT di vari codici componenti “esterni” di uno schema SC

- non esiste più il parametro E_b/N_o : i valori algebrici in ingresso hanno tutti la stessa statistica.
- vari rate e varie memorie
- per $I_a=0$, $I_e=0$. Il contributo estrinseco rimane nullo a lungo (dipende dal rate).
- le curve danno $I_e = 0.5$ per $I_a \approx R$
- di nuovo, una grande memoria del codice è controproducente a basse I_a , poi fa sì che I_e aumenti più rapidamente.

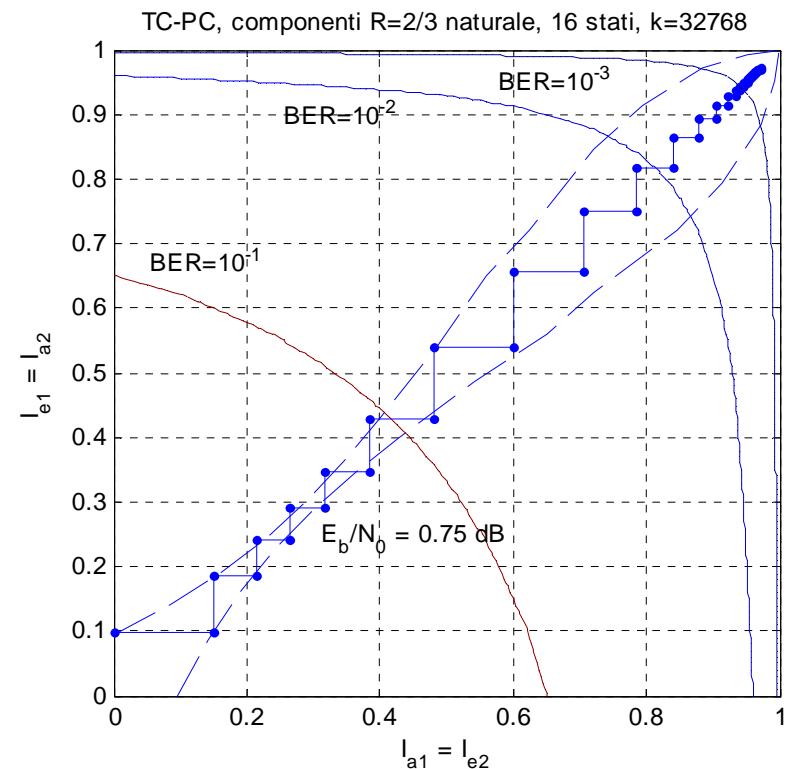


EXIT Chart (1)

Evoluzione dell'informazione mutua dei valori algebrici estrinseci con le iterazioni: se immaginiamo di avere un blocco infinito, i valori algebrici in ingresso sono sempre indipendenti, e quindi a fronte di una certa $I(X,L_{e1})$ che diventa $I(X,L_{a2})$ ci aspettiamo in uscita la $I(X,L_{e2})$ prevista dalla caratteristica EXIT. E così via...

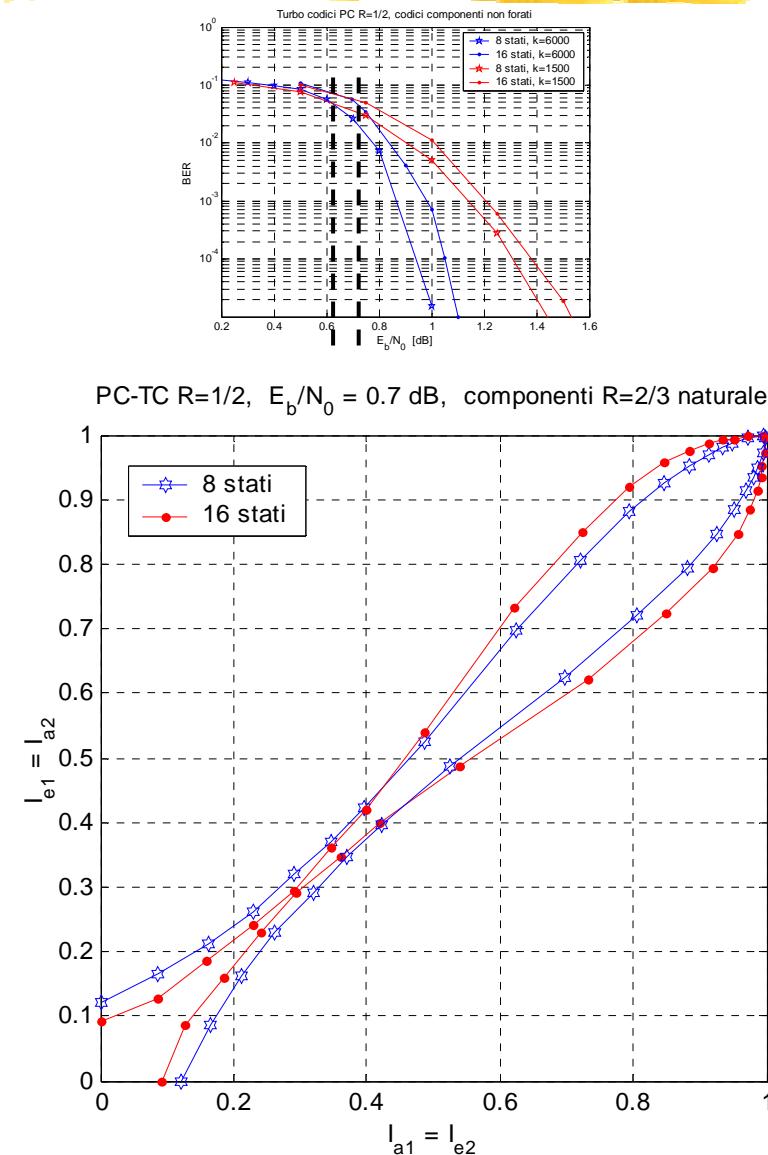
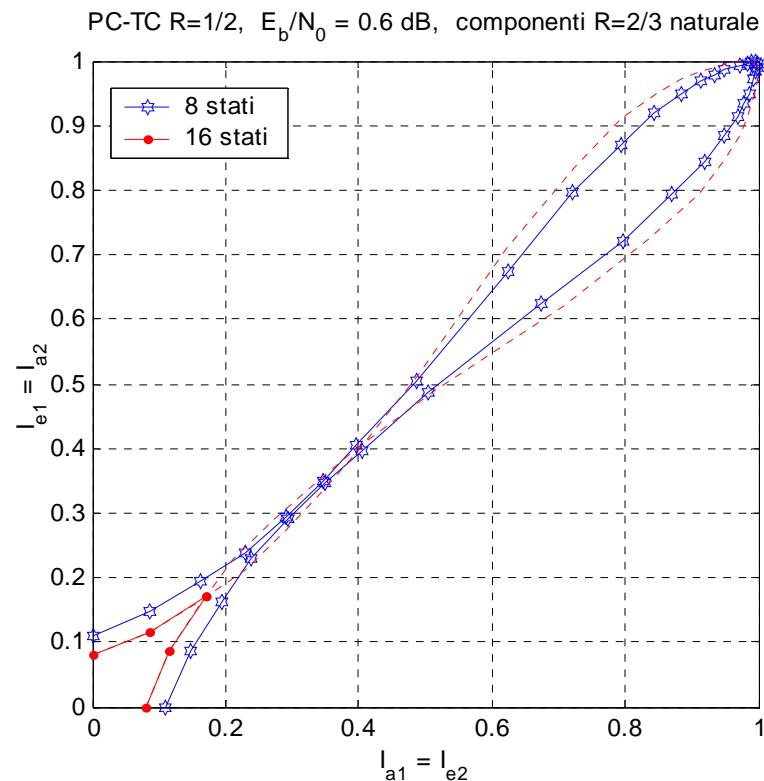
Il comportamento reale si avvicina sempre più a quello teorico all'aumentare della dimensione del blocco. Per blocchi finiti 2 effetti: correlazione ad alte I, e sovrastima della I nel "tunnel".

- soglia: minimo SNR oltre il quale si ha l'apertura del tunnel (con codici componenti uguali, EXIT tangente alla bisettrice)
- si può anche calcolare il BER teorico



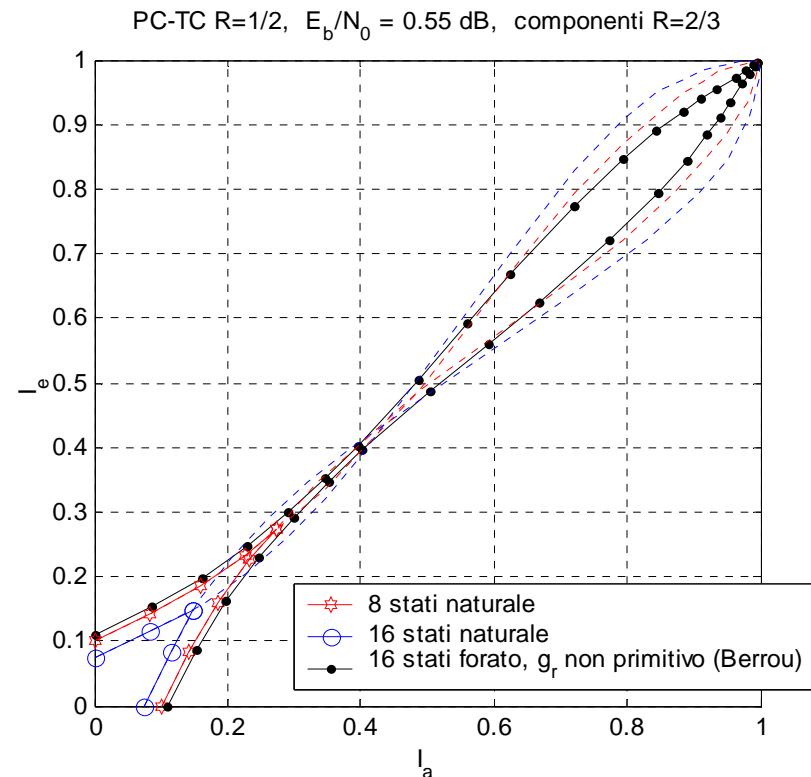
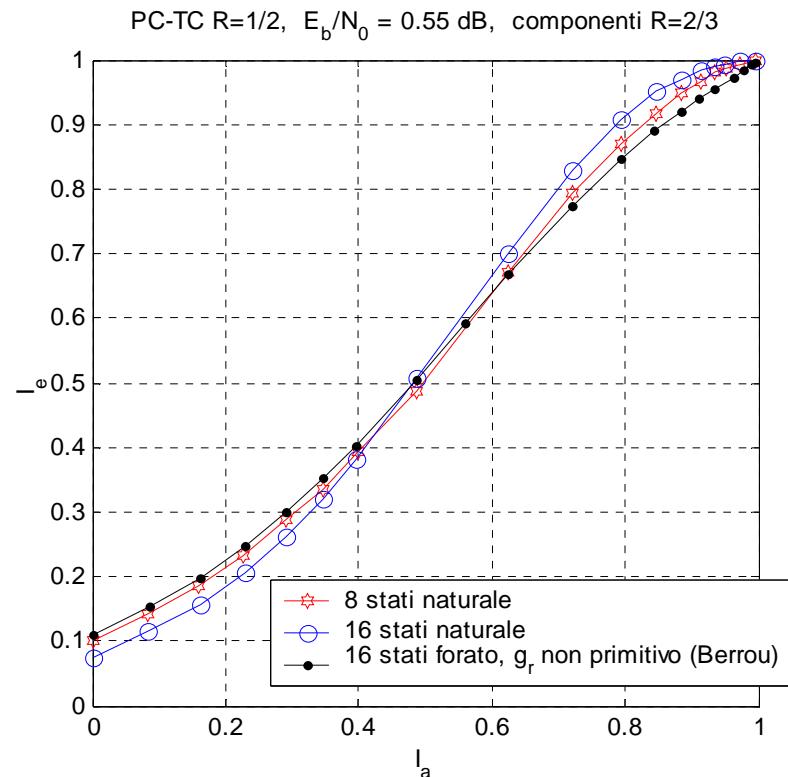
EXIT Chart (2) memoria dei codici componenti

- Confronto soglie sulla EXIT chart



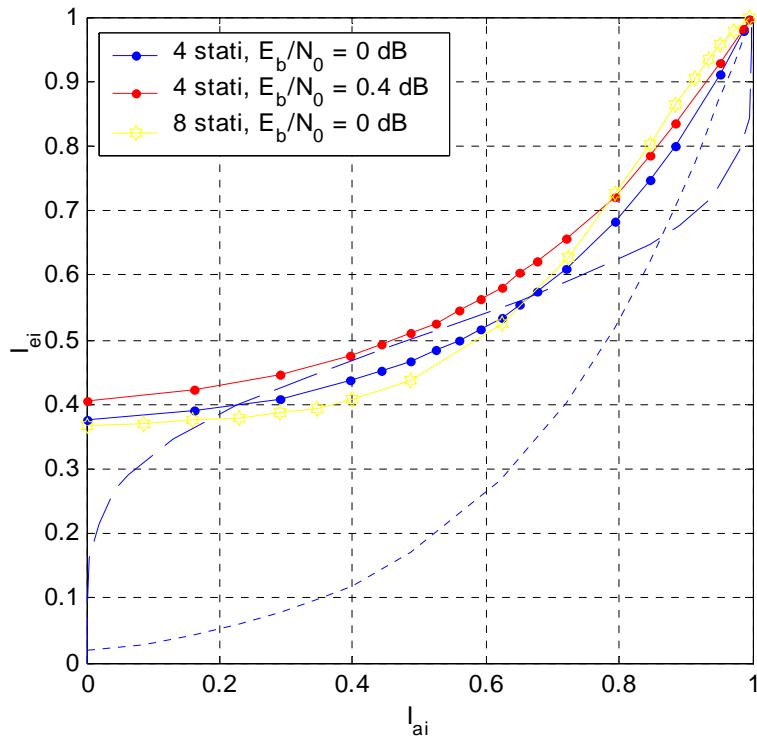
EXIT Chart (3) polinomi

Generatori con polinomio di retroazione primitivo o non primitivo

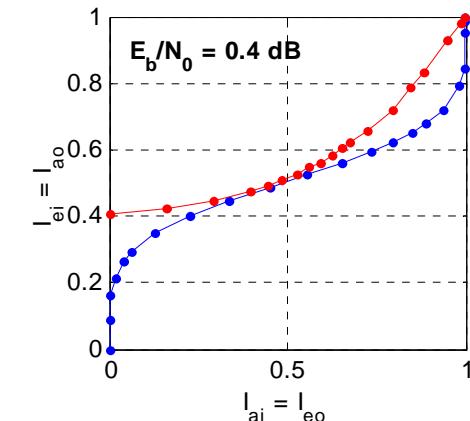
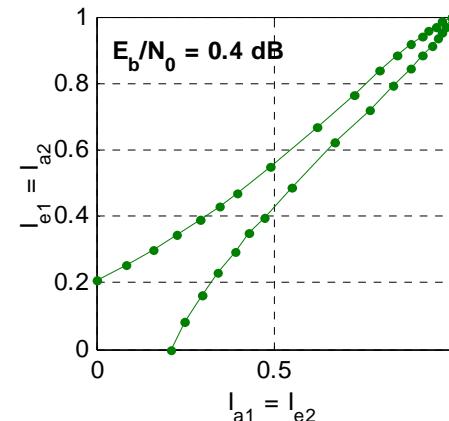
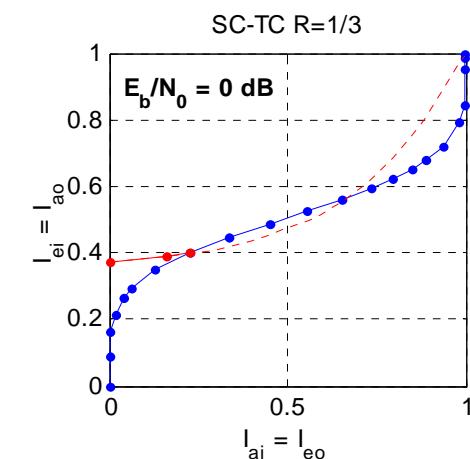
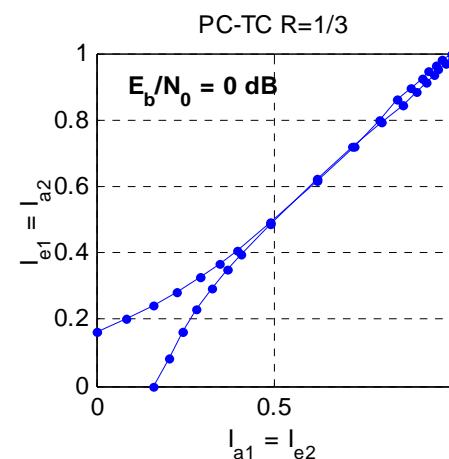
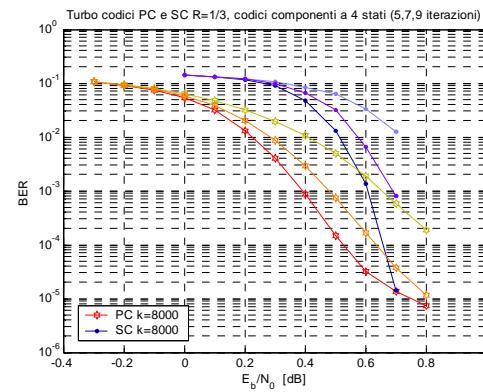


- Esigenze di convergenza contrastanti con esigenze di grande d_2

EXIT Chart (3) PC vs SC



Codici $R=1/3$. La I_e del decodificatore interno comprende anche l'informazione di canale (contributo prevalente a basse I_e => curva quasi piatta)



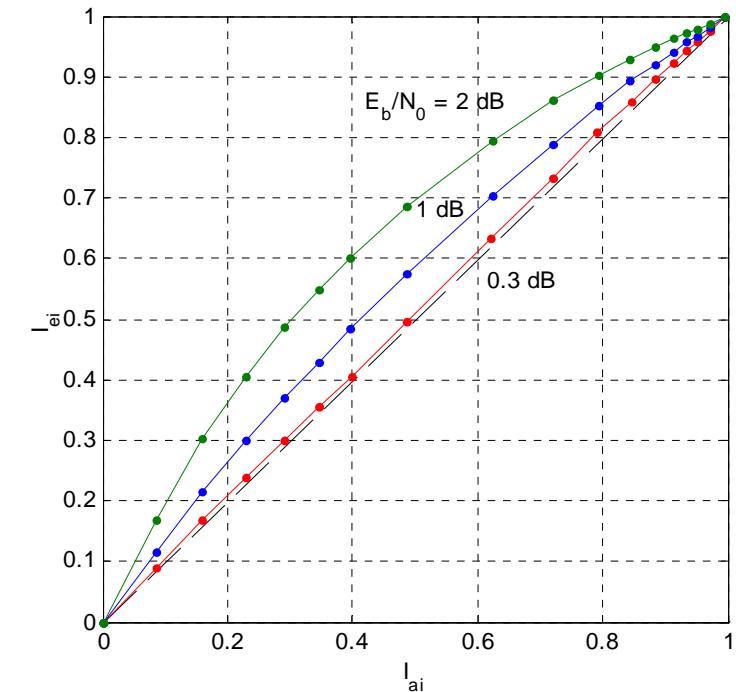
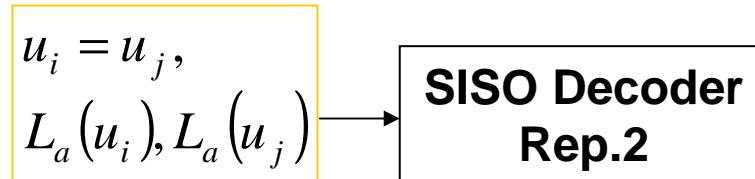
Altri codici Turbo-like (1)

Concentriamoci sull'EXIT chart di un codice concatenato in serie con $R=1/2$:

- R_o deve essere il più basso possibile: al limite $R_o=R=1/2 \Rightarrow R_i=1$. Il codice interno è un accumulatore, p.e.:

$$x_i(D) = \frac{D + D^2 + D^3}{1 + D + D^2 + D^3} u_i(D)$$

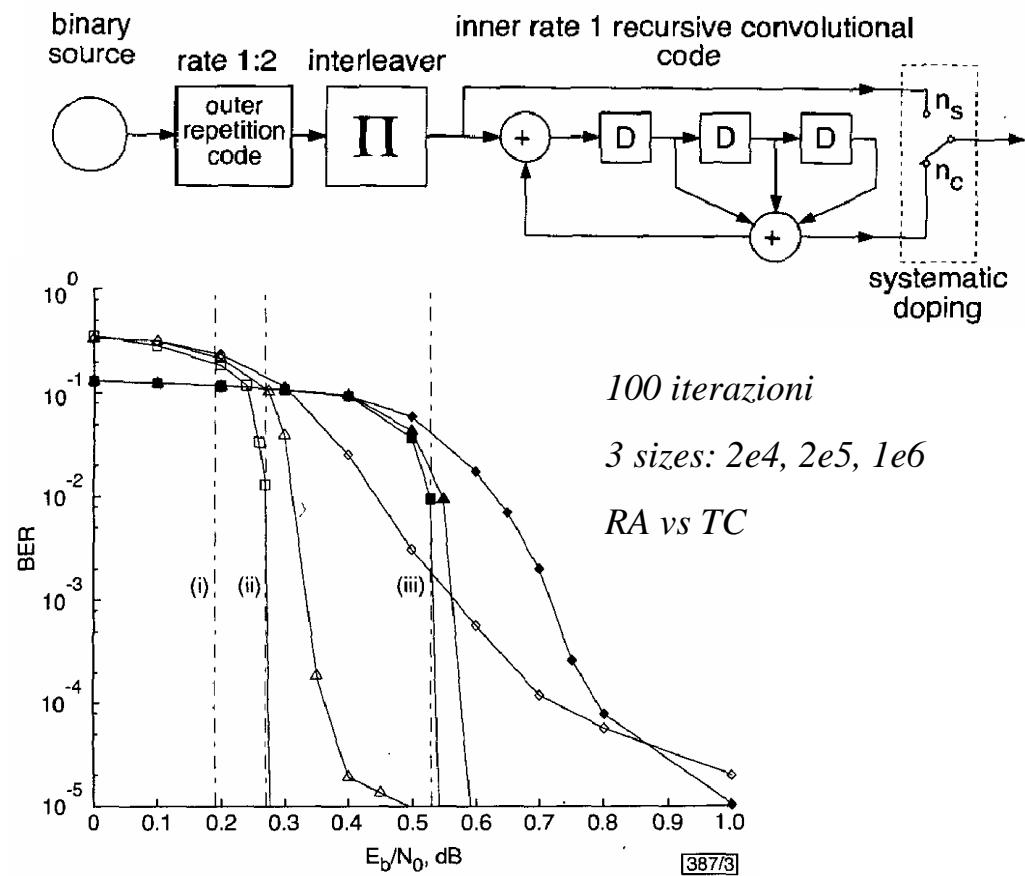
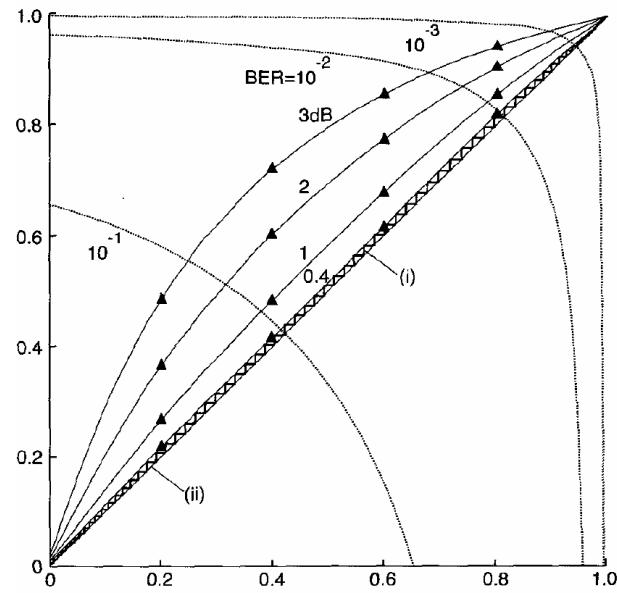
- il codice esterno deve essere il più semplice codice a $R=1/2$ che può venire in mente: codice a ripetizione 2!



- manca l'innesto della decodifica iterativa perchè $I_e=0$ per $I_a=0$ nel codice interno, perchè non c'è ridondanza.

Altri codici Turbo-like (2)

- Repeat accumulate doped codes [tBr99]
- notare il “systematic doping”, indispensabile per far partire la decodifica iterativa.



[tBr99] S. ten Brink, “Rate $\frac{1}{2}$ code for approaching the Shannon limit by 0.1 dB”, Electronics Letters, Jul. 2000, vol.36, No.15, pp. 1293-1294.



Codici di parità a bassa densità (LDPC)

- Storia: Proposti nel 1960 da Gallager [Gal62], riscoperti negli anni '90 dopo i Turbo-codici.
- Codice a blocco (n, K) specificato da una matrice di controllo di parità H sparsa, con n colonne ed m righe ($m = n j / k$ se si impone che ogni riga ed ogni colonna contengano rispettivamente k e j "1"):

ogni riga è associata ad un vincolo di parità e i k uni marcano i bit che partecipano a tale equazione

ogni colonna è associata ad un bit di codice, e i j uni marcano le equazioni di parità a cui tale bit partecipa

Es:

$$(n, j, k) = (10, 2, 4) \quad H = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

[Gal62] R.G. Gallager, "Low-Density Parity-Check Codes", IRE Trans. on Inf. Theory, IT-8, Jan. 1962, pp. 21-28



CNR-IEIIT – Politecnico di Milano



Univ. degli studi di BS

Codici LDPC: proprietà

- Solo $m' < m$ righe di H sono indipendenti: m' degli n bit sono vincolati, i bit di informazione liberi rimangono $K = m' - n$ ed il rate del codice è:

$$R = 1 - \frac{m'}{n}$$

- Se j e k sono costanti per ogni riga e colonna il codice si dice regolare
- Le scelte migliori dei parametri per una buona decodifica sono $j=3$, k di conseguenza $3/(1-R)$ (circa).
- Il massimo numero di uni in comune a due righe, o due colonne si chiama fattore di sovrapposizione e deve essere il più piccolo possibile.
- Per la codifica si può ottenere una matrice di parità equivalente per un codice sistematico, da cui ricavare una matrice generatrice.

Ese:

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \xrightarrow{\text{P}} \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \xrightarrow{\text{PT}} \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$
$$H = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}, \quad G = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$



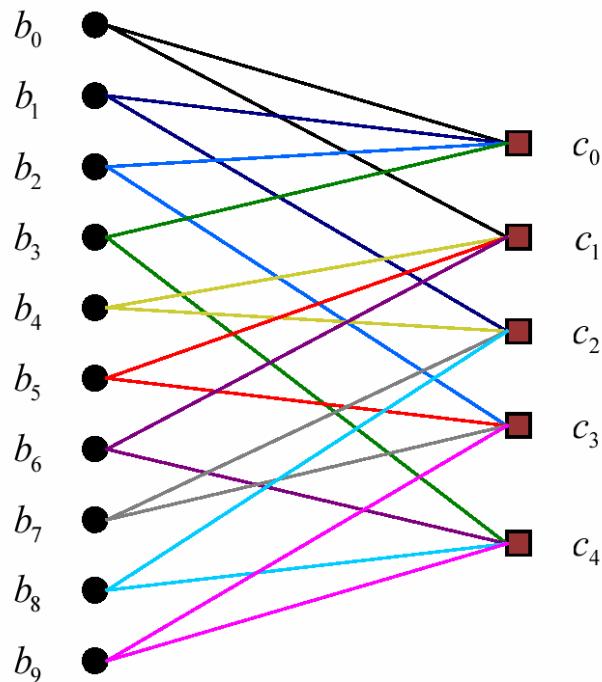
Codici LDPC: decodifica (1)

Ad un codice specificato dalla matrice H si associa il grafo di Tanner:

Esempio: dalla matrice di prima

$$(n, j, k) = (10, 2, 4)$$

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$



Si può vedere come la concatenazione parallela di 5 codici (vincoli) a parità semplice => turbo-decodifica:

- decodifica SISO dei vincoli
- scambio d'informazione estrinseca
- iterazione

Decodifica SISO di codici a parità semplice

Dai valori algebrici indipendenti in ingresso (canale + a priori): $L(x_i)$

Tramite algoritmo di Hartmann e Rudolph [Bel03], poiché il codice duale di un codice SPC ha solo due parole con bit tutti dello stesso segno:

$$L_p(x_i) = L(x_i) + \log \left(\frac{1 + \prod_{l \neq i} \tanh(L(x_l)/2)}{1 - \prod_{l \neq i} \tanh(L(x_l)/2)} \right) \Rightarrow L_e(x_i) = t^{-1} \left(\prod_{l \neq i} t(L(x_l)) \right) \quad (*)$$

Con $t(\cdot) \stackrel{\Delta}{=} \tanh\left(\frac{\cdot}{2}\right)$. Semplificazione: $L_e(x_i) \approx \prod_{l \neq i} \text{sign}(L(x_l)) \cdot \min_{l \neq i} \{|L(x_l)|\}$

Nota: l'espressione (*) si ricava anche senza ricorrere ad Hartmann e Rudolph, osservando che:

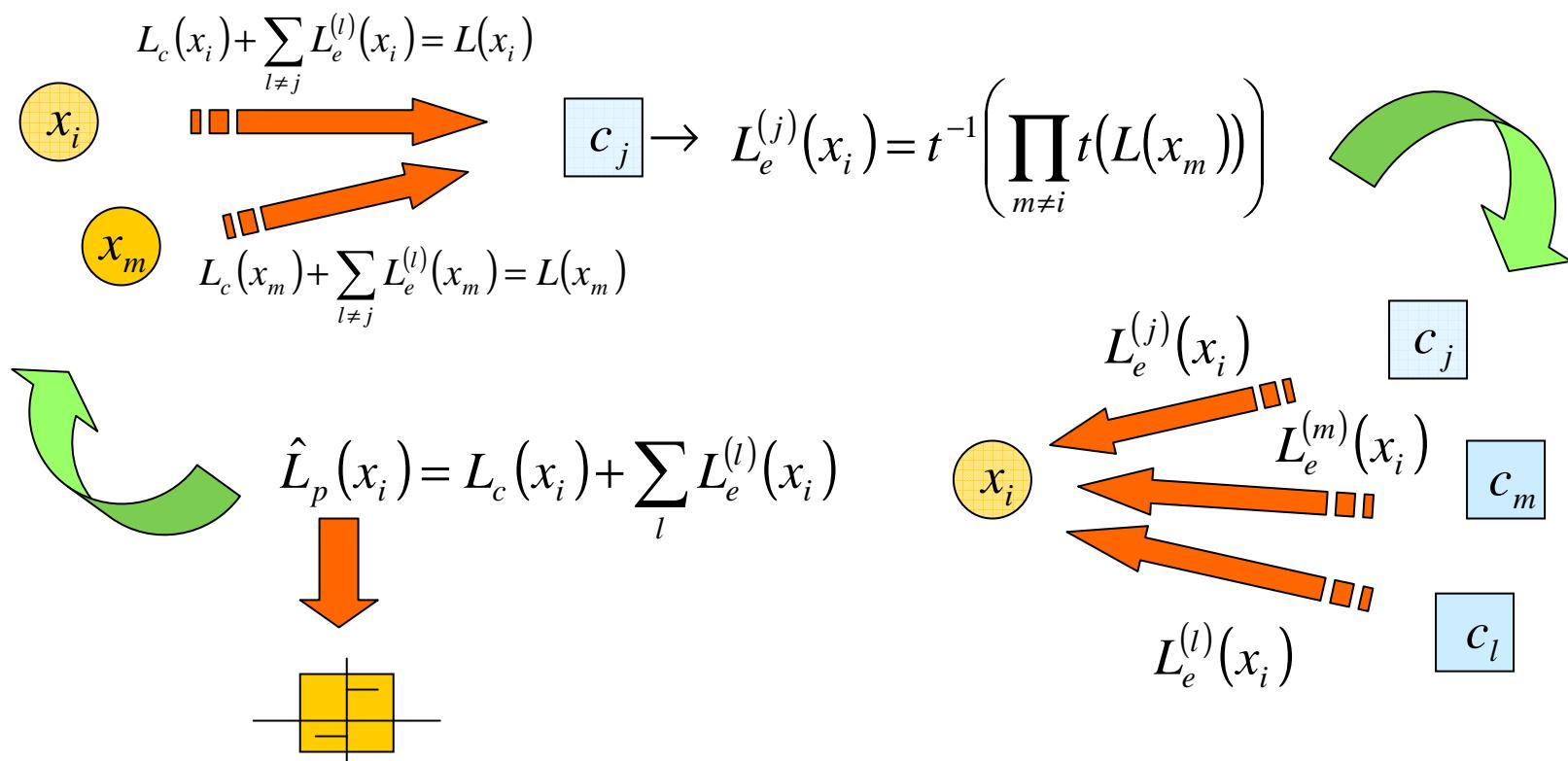
$$\tanh(L(x_l)/2) = P(x_l = +1) - P(x_l = -1) \dots$$

[Bel03] S. Bellini, "Teoria dell'Informazione e codici", www.elet.polimi.it/upload/bellini/tinfcod_c/tinfcod_c.html



Codici LDPC: decodifica (2)

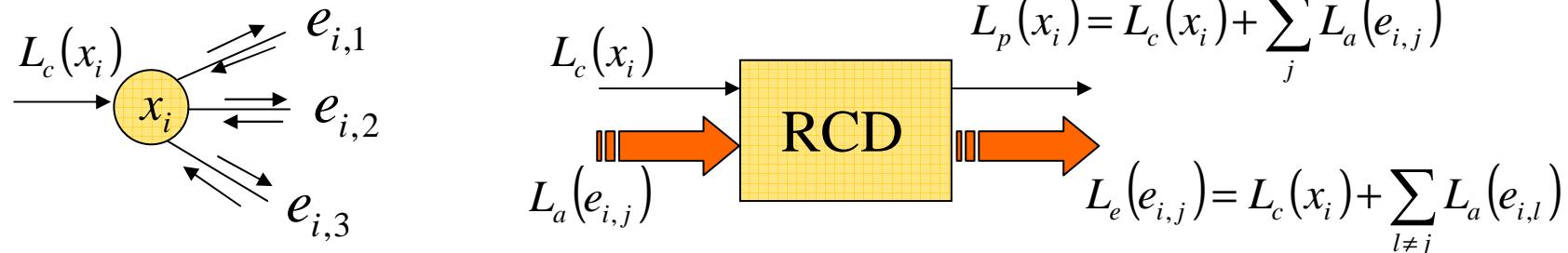
Ad ogni iterazione, ogni nodo di controllo di parità (o di check) può calcolare un contributo estrinseco $L_e^{(j)}$ per ogni bit x_i che vi partecipa, che viene usato all'iterazione successiva come informazione a priori su x_i da ogni altro nodo di controllo di parità che coinvolge x_i .



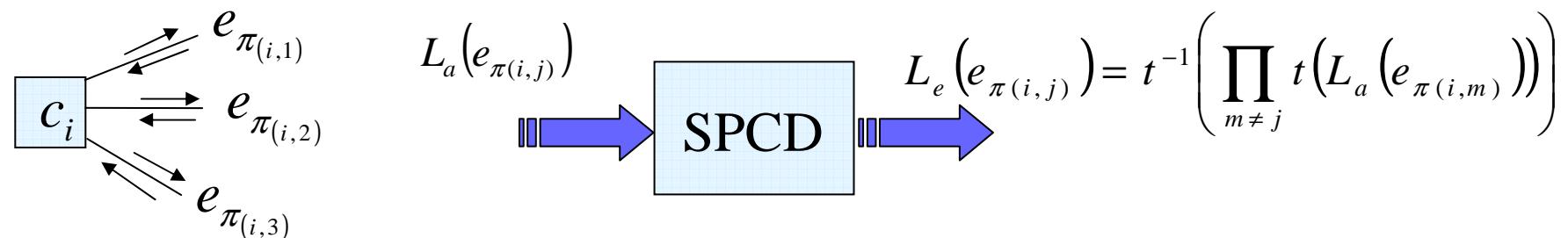
Codici LDPC: analisi della convergenza (1)

Analisi EXIT della convergenza: possiamo vedere il decodificatore LDPC come una coppia di decodificatori SISO che si scambiano informazione soft sui bit e_i dei rami (edges). L'estrinseco dell'uno, diventa *a priori* per l'altro, e l'ordine con cui i rami in uscita dai nodi variabile vengono connessi ai nodi check è gestito da un interleaver π .

Decodificatore di un codice a ripetizione (RCD):

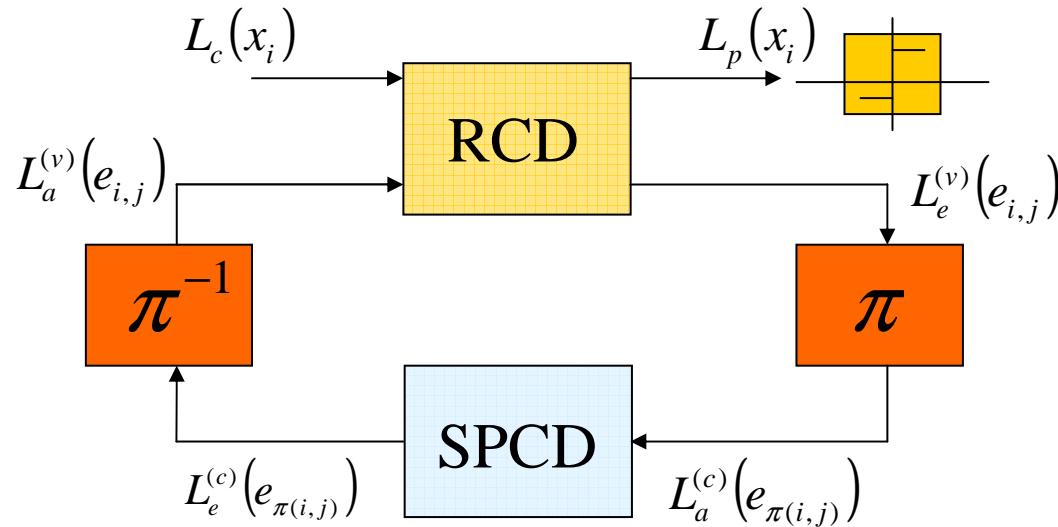


Decodificatore di un codice a parità semplice (SPCD):



Codici LDPC: analisi della convergenza (2)

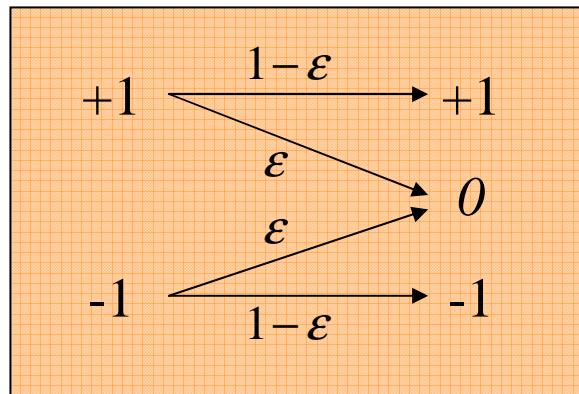
I due decodificatori SISO si scambiano informazione sui bit e_i dei rami:



$$L_e^{(c)}(e_{\pi(i,j)}) = t^{-1} \left(\prod_{m \neq j} t(L_a^{(c)}(e_{\pi(i,m)})) \right) \quad L_e^{(v)}(e_{i,j}) = L_c(x_i) + \sum_{l \neq j} L_a^{(v)}(e_{i,l})$$

Analisi del decodificatore iterativo tramite curve EXIT di ogni decodificatore, calcolate assumendo LLR a priori indipendenti all'ingresso...

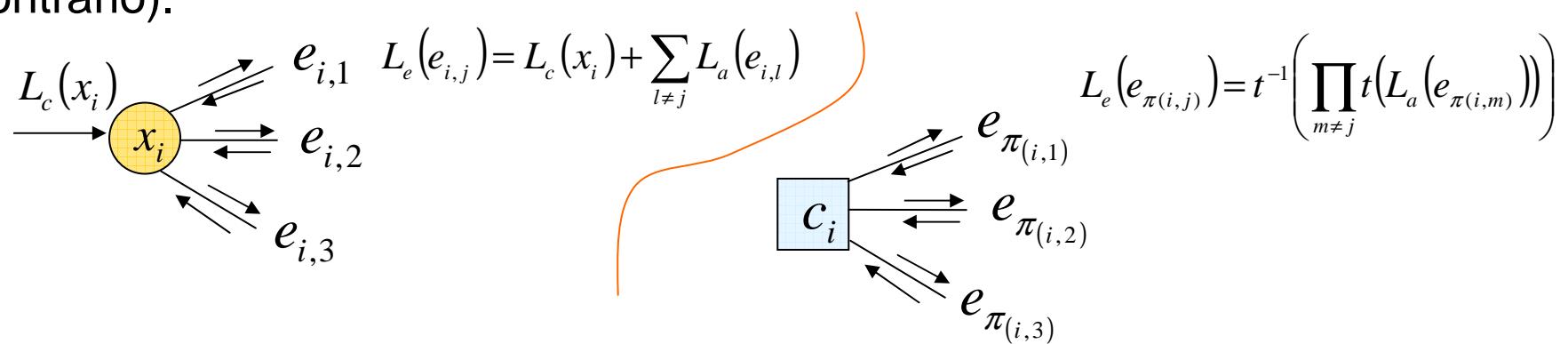
Canale binario con cancellazioni (BEC)



Si può vedere come un canale gaussiano in cui gli LLR di canale sono nulli (incertezza totale, bit cancellato) o infiniti (bit ricevuto corretto).

$$C = H(X) - H(X | Y) = 1 - \varepsilon$$

Un grafo di Tanner con L_c di tipo BEC propaga solo messaggi di tipo BEC. Infatti, al passo zero, i nodi variabile propagano sui rami i messaggi ricevuti da canale. I nodi di check propagano 0 (se c'è almeno un messaggio nullo in ingresso) o il bit corretto di parità (in caso contrario).



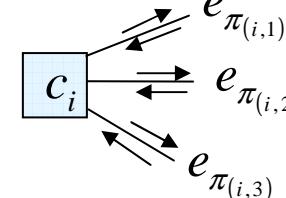
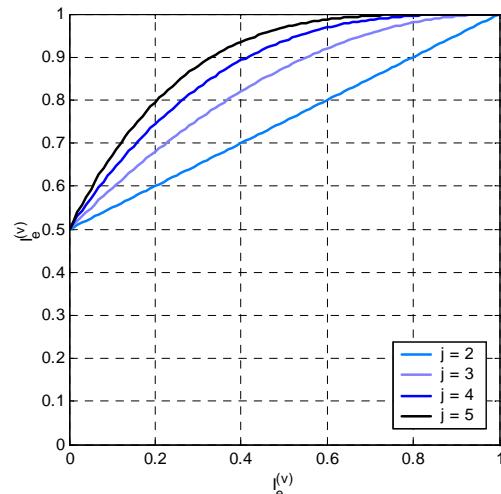
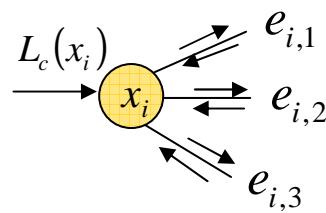
Informazione mutua di messaggi da BEC

L'informazione mutua tra un bit X ed un messaggio Y a lui relativo (una sua osservazione) proveniente da canale BEC è $1 - P[Y=0]$. $\Rightarrow I(L_c, x) = 1 - \varepsilon$

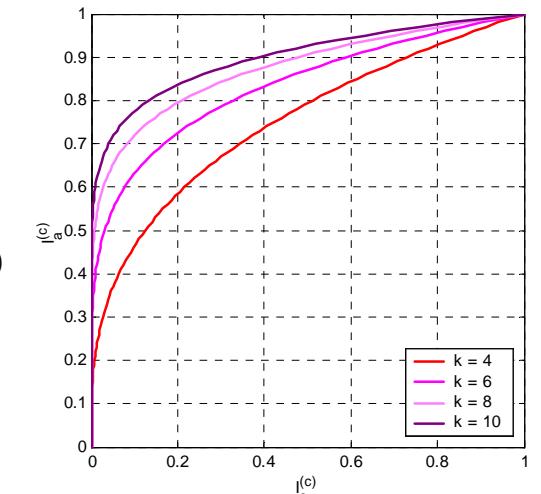
I messaggi a priori sono nulli con probabilità q che facciamo variare a piacere: $\Rightarrow I(L_a, x) = I_a = 1 - q$

Per i messaggi estrinseci vale:

$$P[L_e(e_{i,j}) = 0] = \varepsilon q^{j-1} \Rightarrow I_e^{(v)} = 1 - \varepsilon (1 - I_a)^{j-1}$$



$$P[L_e(e_{i,j}) \neq 0] = (1 - q)^{k-1} \Rightarrow I_e^{(c)} = I_a^{k-1}$$



Analisi della convergenza tramite EXIT chart (1)

La decodifica iterativa converge se c'è un tunnel aperto tra le due curve EXIT.

Invertendo la EXIT del SPCD si ottiene:

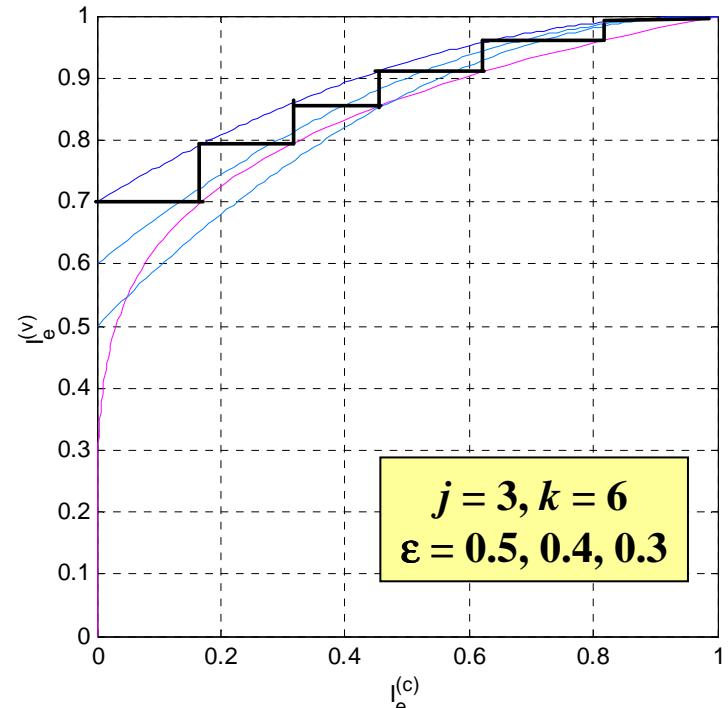
$$f(x) = 1 - \varepsilon(1-x)^{j-1} - x^{\frac{1}{k-1}} > 0 \quad \forall x \in [0,1)$$

Possiamo chiederci

1. qual'è il punto più critico?
2. quale soglia ε_T (il massimo ε) sotto il quale il tunnel è aperto?
3. dato R (p.e. $\frac{1}{2}$) quali j, k massimizzano ε_T ?

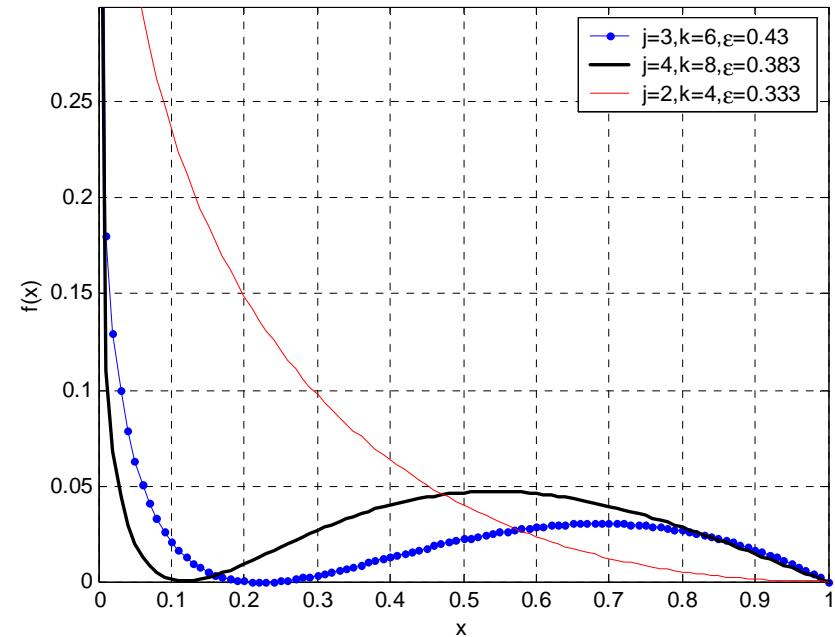
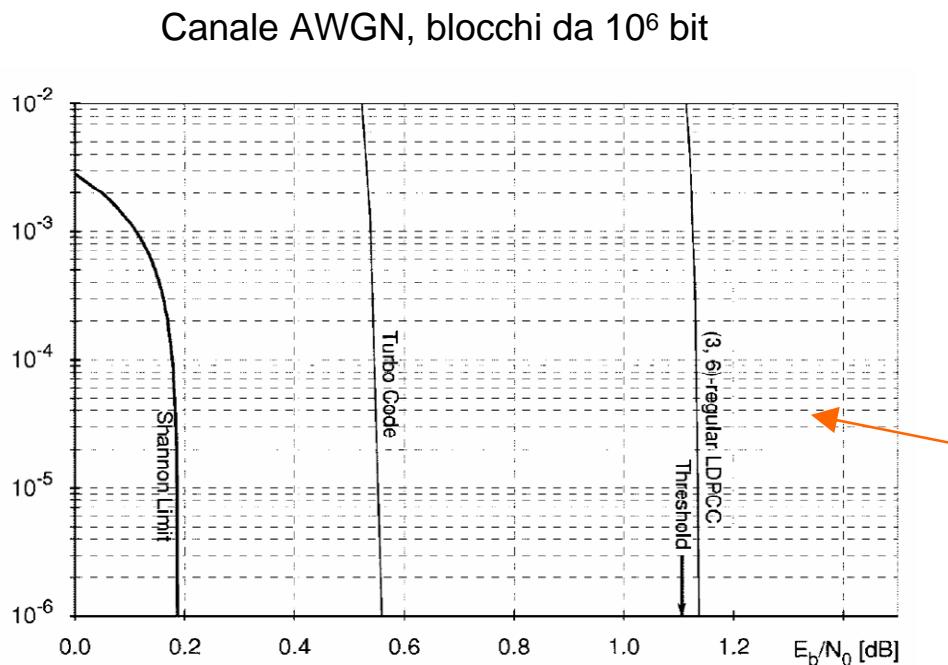
Il punto più critico è intorno a $I_a=0.1-0.2$, ma attenzione nell'intorno di $I_a=1$! Per avere il tunnel intorno a $I_a=1$ occorre (condizione di stabilità):

$$\varepsilon(j-1)(1-x)^{j-2} \Big|_{x \rightarrow 1} < \frac{1}{k-1} \quad \Rightarrow \quad \varepsilon < \frac{1}{k-1}$$



Analisi della convergenza tramite EXIT chart (2)

Si scopre che la soglia più alta (0.43) si ha per $j=3$, $k=6$. Ma la capacità (0.5) è un po' lontana.



Su canale gaussiano valgono considerazioni simili: la miglior convergenza si ottiene per $j=3$, $k=6$, ma la soglia (1.1 dB), è lontana dalla capacità (0.2 dB).

Codici LDPC irregolari

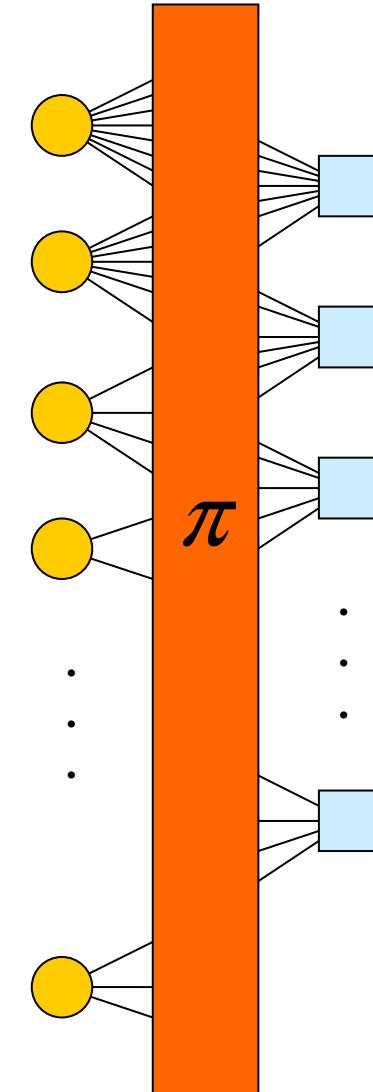
Assumiamo un grafo di Tanner con E rami, e grado dei nodi non costante, per cui una frazione λ_i dei rami è connessa a nodi variabile di grado i , ed una frazione ρ_i è connessa a nodi check di grado i .

$$n = E \sum_i \frac{\lambda_i}{i} = E \int_0^1 \lambda(x) dx \quad \text{con} \quad \lambda(x) = \sum_i \lambda_i x^{i-1}$$

$$m = E \sum_i \frac{\rho_i}{i} = E \int_0^1 \rho(x) dx \quad \text{con} \quad \rho(x) = \sum_i \rho_i x^{i-1}$$

Assumendo che i check siano tutti indipendenti, per R vale la solita relazione, che in funzione di λ e ρ diventa:

$$R = 1 - \frac{\int_0^1 \rho(x) dx}{\int_0^1 \lambda(x) dx}$$



Analisi EXIT di codici LDPC irregolari

Per un messaggio in uscita da un nodo variabile o check di grado i :

$$I_{e,i}^{(v)} = 1 - \varepsilon (1 - I_a)^{i-1} \quad I_{e,i}^{(c)} = I_a^{i-1}$$

L'informazione media in uscita dai nodi è allora:

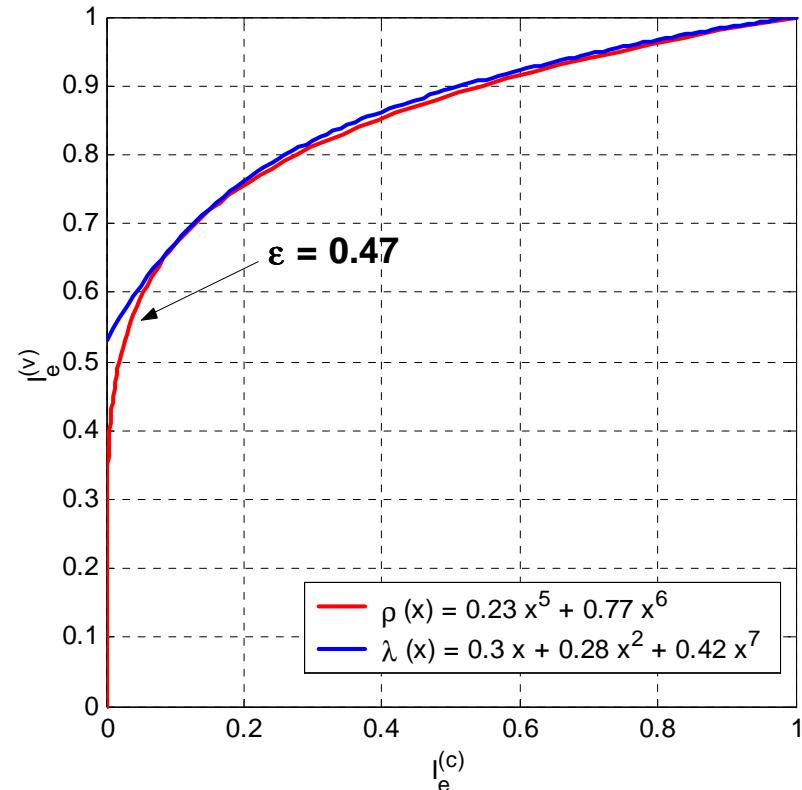
$$I_e^{(v)} = \sum_i \lambda_i I_{e,i}^{(v)} = 1 - \varepsilon \lambda (1 - I_a)$$

$$I_e^{(c)} = \sum_i \rho_i I_{e,i}^{(v)} = \rho(I_a)$$

La condizione per la convergenza della decodifica iterativa diventa:

$$1 - \varepsilon \lambda (1 - x) > \rho^{-1}(x) \quad \forall x \in [0,1]$$

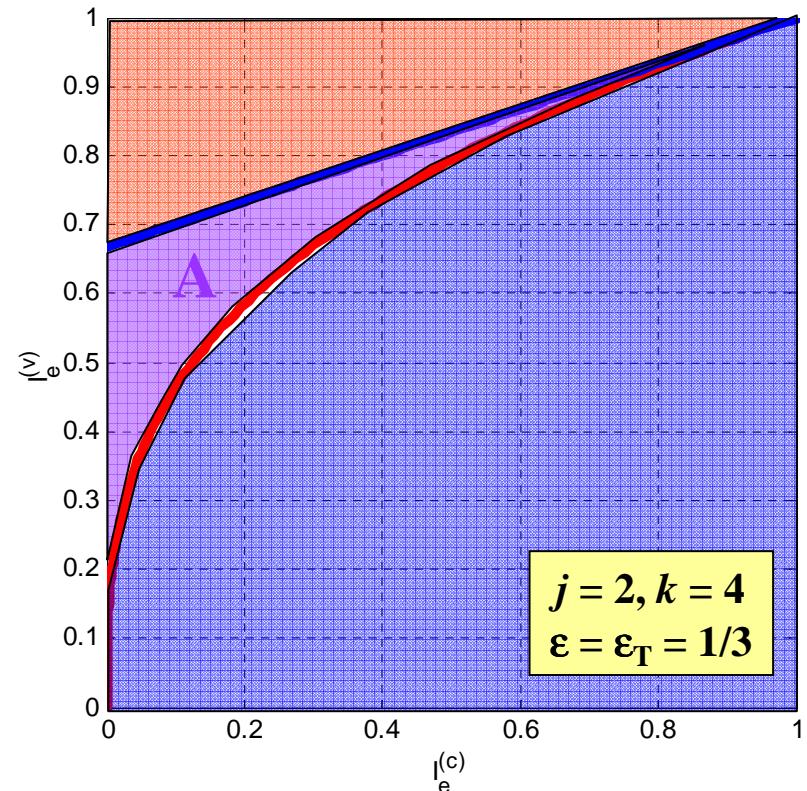
Si modellano le EXIT agendo su λ e ρ a pari R , cercando di alzare la soglia ε_T ...



LDPC irregolari “capacity-achieving”

Le aree sottese dalle curve EXIT sono legate a R. L'area A compresa tra le due curve EXIT vale:

$$\begin{aligned} A &= \int_0^1 I_e^{(v)}(x)dx + \int_0^1 I_e^{(c)}(x)dx - 1 = \\ &= -\varepsilon \int_0^1 \lambda(x)dx + \int_0^1 \rho(x)dx = \\ &= (C - R) \int_0^1 \lambda(x)dx = (C - R) \frac{n}{E} \end{aligned}$$



Se $A<0$ la decodifica iterativa non può convergere, ma ogni $A>0$ per $\varepsilon=\varepsilon_T$ comporta un'intrinseca perdita di rate rispetto alla capacità...

Per essere “capacity-achieving” le EXIT devono coincidere!

Codici LDPC: pregi e difetti

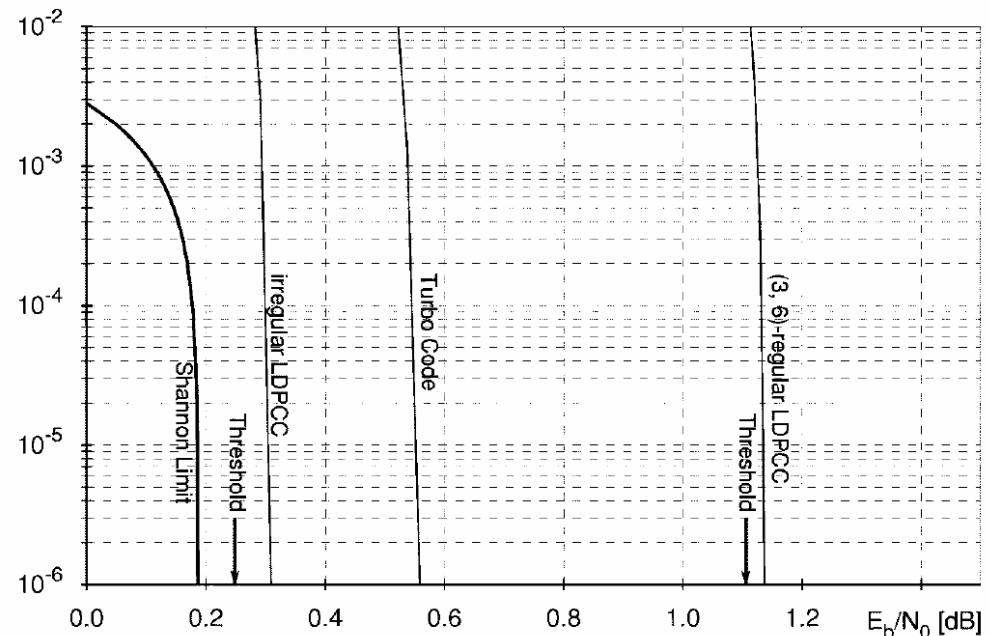
Da [RicShoUrb01]:

$N=10^6, R=1/2$

$\lambda(x)$ di grado 49

$\rho(x)$ di grado 10

soglia a 0.05 dB dalla capacità.



Pregi: semplicità di decodifica SISO, parallelizzabilità (blocchi grandi)

Difetti: richiede più iterazioni per la convergenza, prestazioni asintotiche condizionate dalla d_{min} tra *pseudo-parole di codice*!

[RicShoUrb01] T. Richardson, M. Shokrollahi, R. Urbanke, “Design of Capacity-approaching Irregular Low-Density Parity-Check Codes”, IEEE Trans. on Inf. Theory, vol.47, No.2, Feb. 2001, pp. 619-637



CNR-IEIIT – Politecnico di Milano



Univ. degli studi di BS

Altre applicazioni del “principio turbo”

Stima “quasi MAP” di una sequenza di simboli sottoposta a due diversi set di vincoli, osservata attraverso un canale rumoroso, tramite analisi SISO iterativa con scambio di informazione estrinseca tra i dispositivi che applicano i vincoli (necessaria la presenza di un interleaver tra i due).

- codifica / modulazione (Turbo Demodulazione) [tBrSpY98],[MoqAul01]
- ISI / decodifica (Turbo equalizzazione) [Dou95]
- MU detection / decodifica (Turbo MU detection) [WaPo99]
- MIMO detection / decodifica (Turbo MIMO detection) [ZuWaDeVa03]

[tBrSpY98] S. ten Brink, J. Speidel, R.H. Yan, “Iterative Demapping and Decoding for Multi-level Modulation”, in Proc. GLOBECOM '98 Sydney, Australia, Nov. 8-12, 1998, pp. 579-584

[MoqAul01] P. Moqvist, T. Aulin, “Serially Concatenated Continuous Phase Modulation With Iterative Decoding”, IEEE Trans. on Commun., Vol. 49, No. 11, Nov. 2001, pp. 1901-1915

[Dou95] C. Douillard et al., “Iterative correction of intersymbol interference: Turbo equalization”, Eur. Trans. on Telecomm., Vol. 6, Sept.-Oct. 1995, pp. 507-511

[WaPo99] X. Wang and H.V. Poor, “Iterative (Turbo) Soft Interference Cancellation and Decoding for Coded CDMA”, IEEE Trans. on Commun., Vol. 47, No. 7, Jul. 1999, pp. 1046-1061

[ZuWaDeVa03] D. Zuyderhoff, X. Wautelet, A. Dejonghe, L. Vandendorpe, “MMSE turbo receiver for space-frequency bit-interleaved coded OFDM,” in Proc. of Vehicular Technology Conference, Oct. 2003, pp. 567–571

