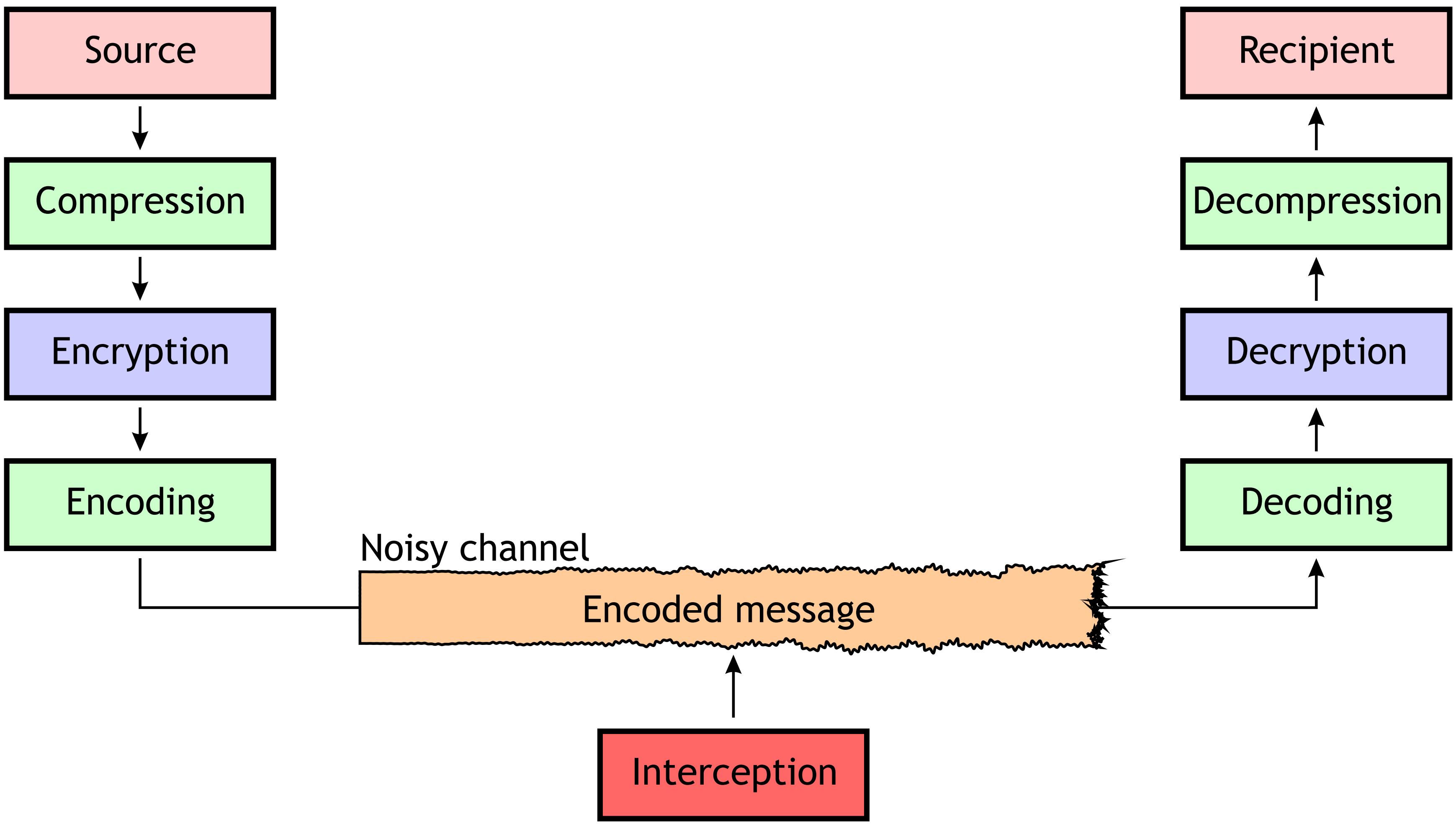
Recovering a code's length and synchronization from a noisy intercepted bitstream.

M. Cluzeau and M. Finiasz

INRIA

ENSTA ParisTech

Overview of the problem



- We intercept a noisy bitstream and want to recover the (encrypted) information.

Overview of the problem

- Most of the time, coding schemes are standardized
  - ▷ no need for code reconstruction.
- Yet, "some people" are interested in this:
  - ▷ not many public works on this topic,
  - ▷ many interesting problems arise, depending on the type of code we focus on.
- Here we focus on linear block codes requiring to:
  - ▷ find the block length,
  - ▷ find a generator/parity check matrix,
  - ▷ find an efficient decoder,

    $\longrightarrow$ we do not address this problem here.

Overview of the problem
The case of linear block codes

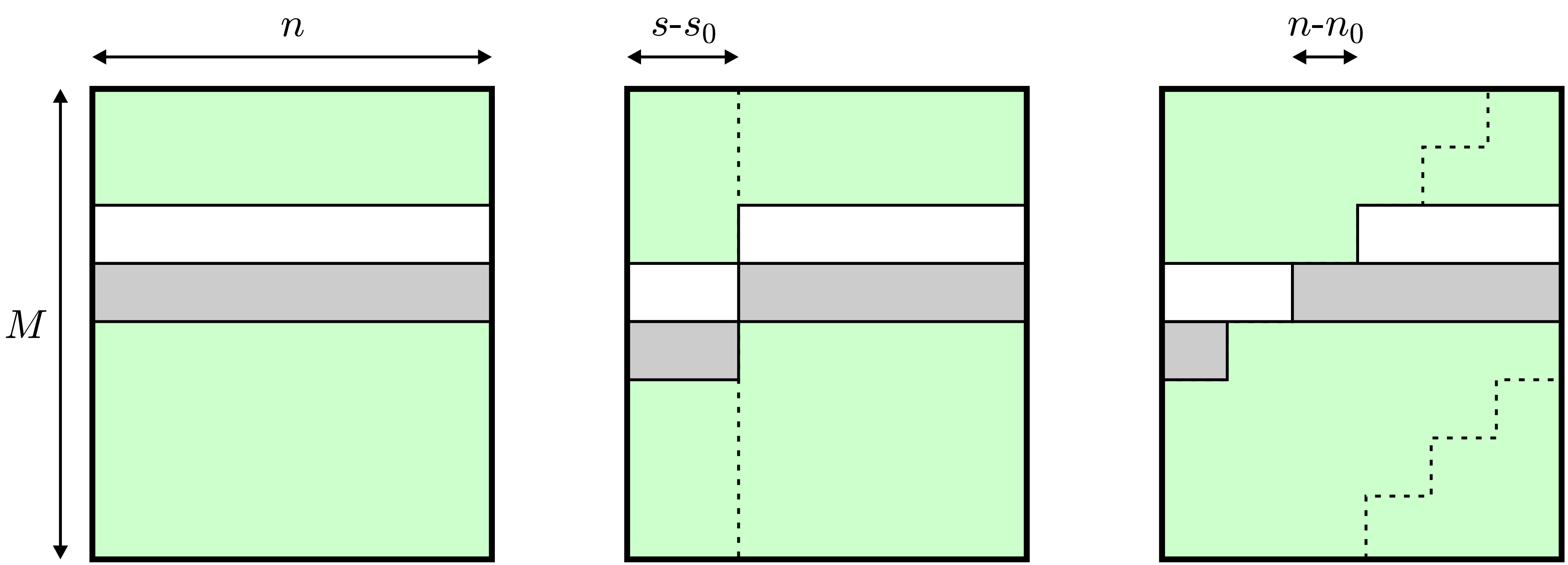
s n

010010 11010000 10111010 10010100 101110...

- The only thing we have is a noisy bitstream:
  - we need to find s_0 and n_0 the synchronization and length of the code.

- For very short codes of small dimension various techniques can give us some hint on n,
  - none of them work for real life codes...
    - → we have to test each choice of s and n.

- For given s and n build the matrix $\mathcal{G}$ of "codewords"
  - if $n = n_0$ and $s = s_0$ it has minimal rank k,
  - if $n = n_0$ and $s \neq s_0$ it has rank $\min(k + |s - s_0|, n)$,
  - if $n \neq n_0$ it has rank n.

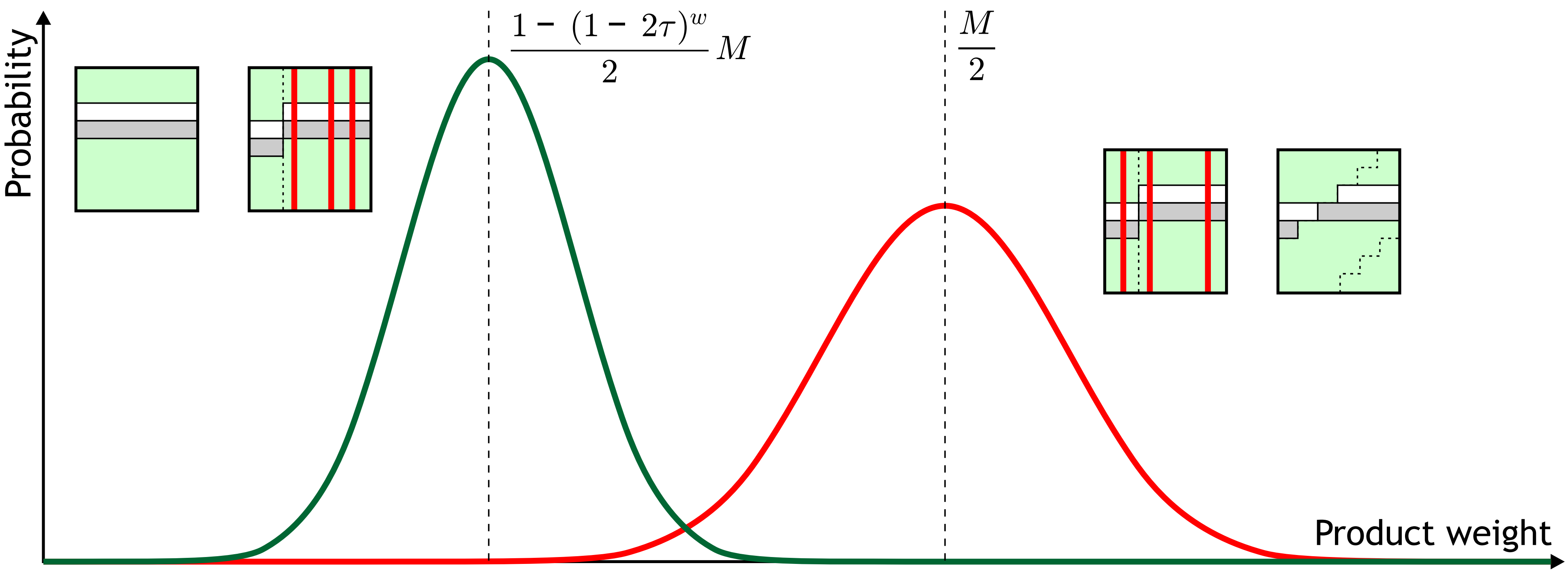


- For given s and n build the matrix $\mathcal{G}$ of “codewords”
  - if $n = n_0$ and $s = s_0$ it has minimal rank k,
  - if $n = n_0$ and $s \neq s_0$ it has rank $\min(k + |s - s_0|, n)$,
  - if $n \neq n_0$ it has rank n.

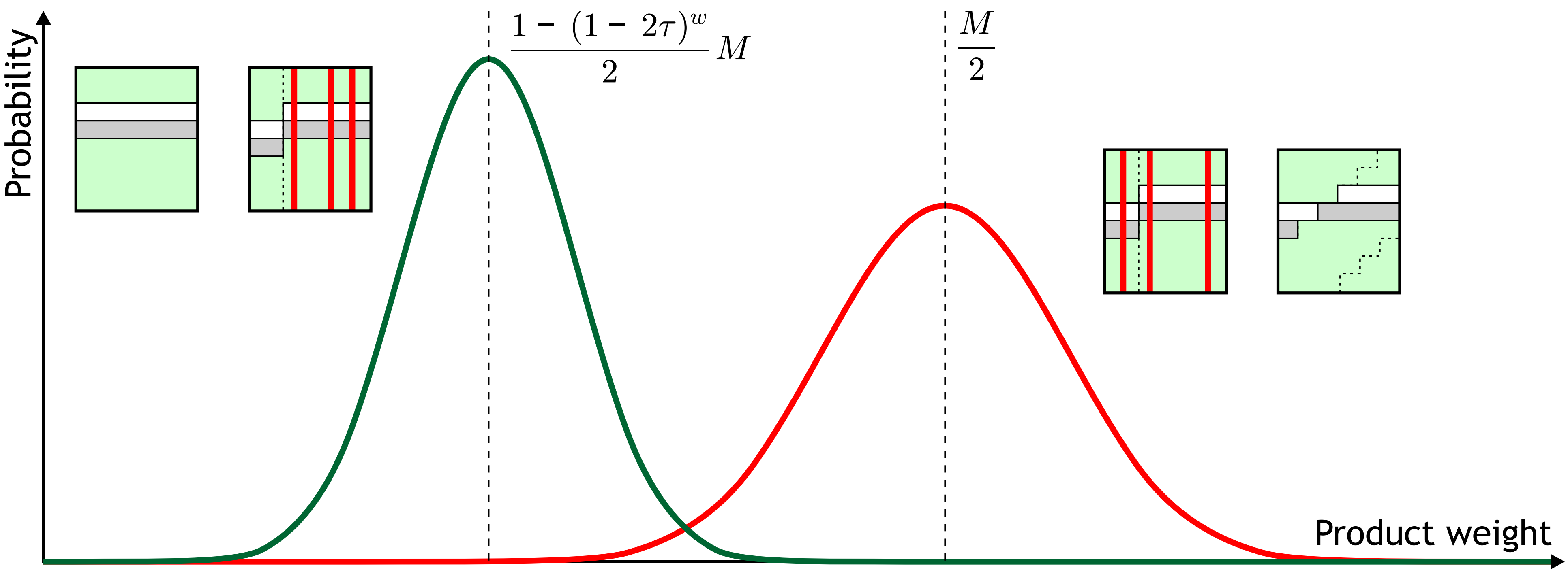
- Very efficient to guess n_0 and then s_0,
  → only for very low noise levels $\tau \ll \frac{1}{n}$.

- For higher noises the rank is always n...

- If n and s are correct, a word of the dual of the target code multiplied by $\mathcal{G}$ should have low weight,
  - suppose we have such a dual word of low weight w.



- If a word following the green distribution is found, $n = n_0$
  - and $s - s_0$ is probably small.



The algorithm we propose

- We need to exhaustively search through the possible s and n.

- Successively go through the possible values of n
  - for each length "test" several synchronizations s $\longrightarrow$ different possible heuristics.

- Testing a pair (n, s) consists in searching for a dual word following the green distribution:
  - exhaustive search of words of given weight
  - using Valembois' algorithm.

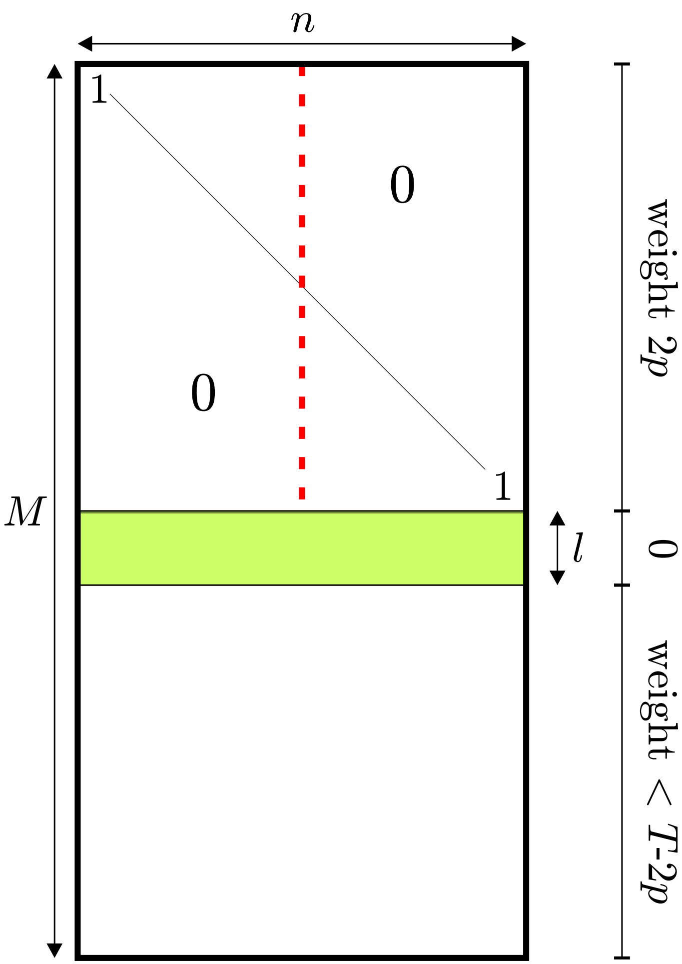
Exhaustive search of given weight dual words

- We look for a dual word of length n and weight w.

- We can find all such dual words using:
  - straight-forward exhaustive search
    $\longrightarrow$ $O(n^w)$ time and $0(1)$ memory.
  - the birthday algorithm
    $\longrightarrow$ $O(n^{\frac{w}{2}})$ time and $O(n^{\frac{w}{2}})$ memory.
  - the Chose-Joux-Mitton algorithm [Eurocrypt 2002]
    $\longrightarrow$ $O(n^{\frac{w}{2}})$ time and $O(n^{\lceil\frac{w}{4}\rceil})$ memory.

- Very efficient for codes with very low weight dual words
  $\longrightarrow$ typically LDPC codes.

Valembois' algorithm



- ▷ Based on the Canteaut-Chabaud decoding algorithm,
- ▷ does not focus only on low weight dual words,
- ▷ small memory requirements.

- Very efficient for low noise levels,
  → tolerates higher noise levels for very short codes.

- Codes of rate $\frac{1}{2}$:
  - no low weight dual words,
  - for our problem: among the difficult cases.

- Dual words found in 10000 iterations of Valembois' algorithm (less than a second).

n \ τ	0.001	0.002	0.005	0.01	0.02	0.05
32	14637	27081	42570	42913	19464	210
64	∞	∞	∞	1172189	6310	0
128	∞	∞	∞	2992	0	0
256	∞	∞	0	0	0	0

- LDPC codes of rate $\frac{1}{2}$ and weight 6 parity checks,
  - find words for lengths up to 10000 with 2GB memory.

- For an LDPC of length 1000 in 50 iterations ($\sim$2 min.)

τ	words found	expected words per iteration	expected total words found
0.01	478	41	492
0.02	251	7.5	266
0.03	84	1.5	70
0.04	15	0.33	16
0.05	6	0.08	3.9
0.06	1	0.02	1.0

- We can find the length/synchronization of a code by using reconstruction techniques,
  - easier for codes with low weight dual words $\longrightarrow$ LDPC
  - not very satisfying for random codes.

- For an unknown code, both techniques should be tried
  - for very low noise levels, Valembois' algorithm is faster, even for long LDPC codes.

- For other kind of codes:
  - convolutional codes [Côte,Sendrier - ISIT09]
  - turbocodes $\longrightarrow$ we are working on it...