Turbo Codes for Deep Space Communications: CCSDS 131.0-B-2 standard implementation

Final project for the Channel Coding course

Gianluca Marcon
University of Padova
gianluca.marcon.1@studenti.unipd.it

August 4, 2017

Standard specifications

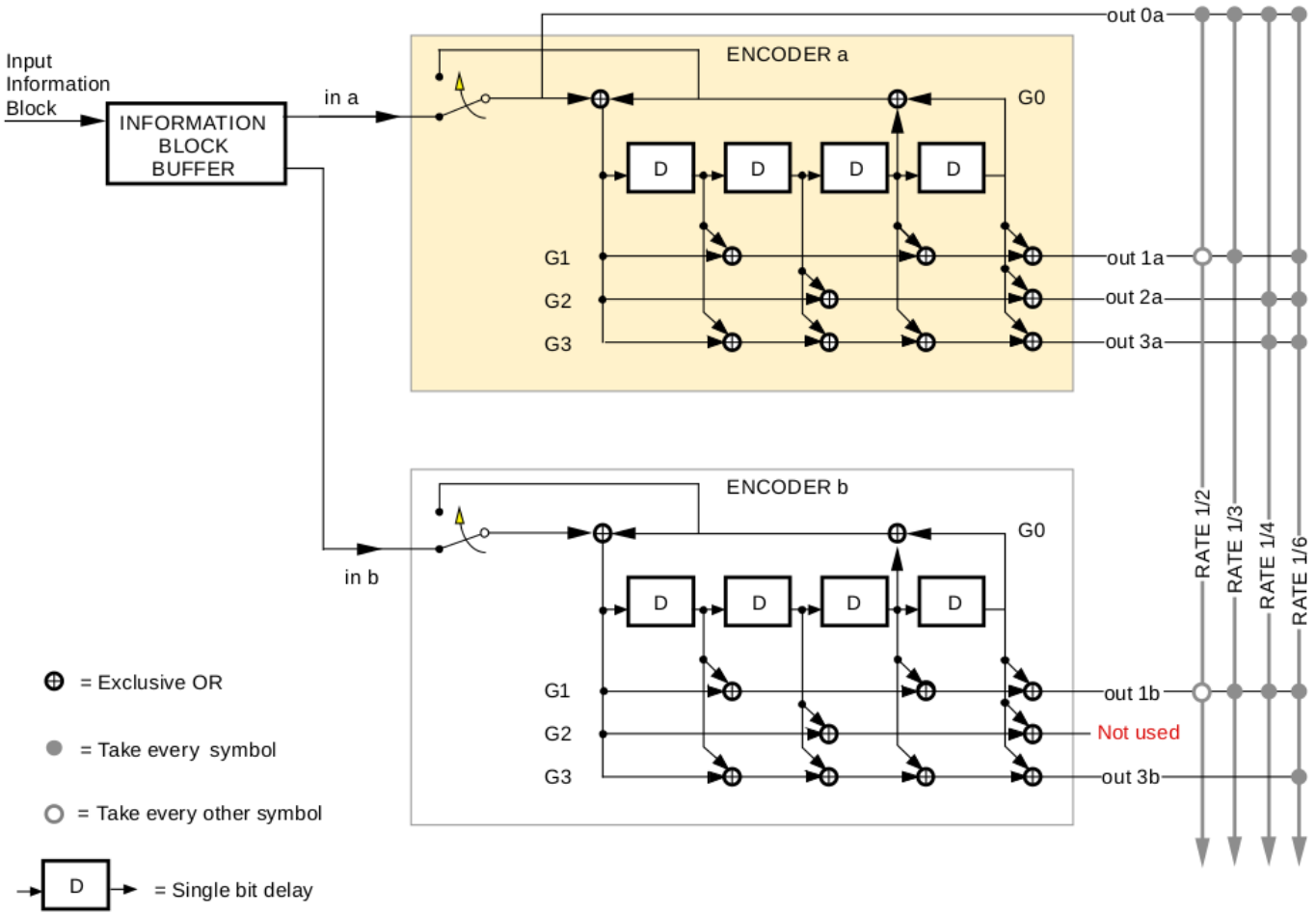
The standard specifies different input packet lengths k

- 1784
- 3568
- 7136
- 8920

...and different code rates R

- $1/2$
- $1/3$
- $1/4$
- $1/6$

Encoder structure

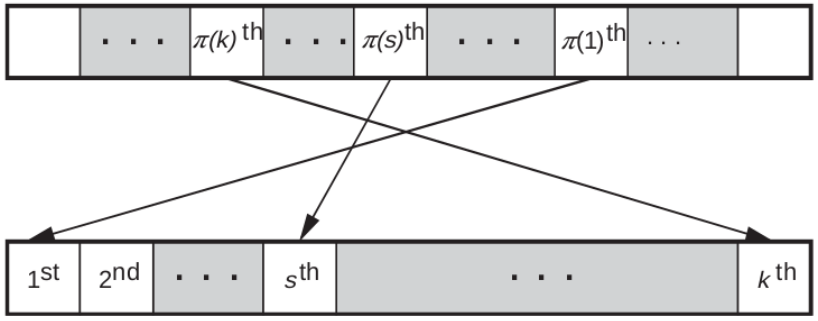


Convolutional codes defined through forward and backward connection vectors

Example: defining a code in C

```
// define first code
int N_components = 2;
char *forward[N_components];
forward[0] = "10011";
forward[1] = "10101";

char *backward = "0011";

t_convcode code = convcode_initialize(forward, backward,
                                      N_components);

t_turbocode turbo = turbo_initialize(code, code, pi,
                                     info_length);
```

Interleaver



i-th bit of the interleaved packet is the $\pi(i)$-th bit of the original packet

Input length k	$k_1 \times k_2 \times k_3$
1784	$8 \times 223 \times 1$
3568	$8 \times 223 \times 2$
7136	$8 \times 223 \times 4$
8920	$8 \times 223 \times 5$

Building the interleaver

$$p = \begin{bmatrix} 31 & 37 & 43 & 47 & 53 & 59 & 61 & 67 \end{bmatrix}$$

for $s = 1$ to k do

$$m = (s-1) \mod 2$$

$$i = \text{floor}\,[(s-1)/(2k_2)]$$

$$j = \text{floor}\,[(s-1)/2] - ik_2$$

$$t = (19i+1) \mod (k_1/2)$$

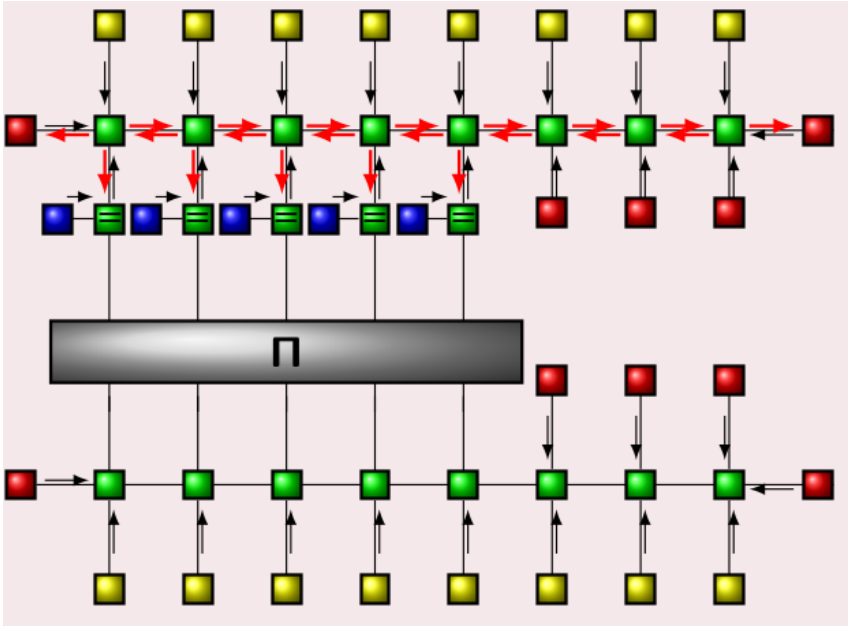
$$q = t \mod 8 + 1$$

$$c = (p_q j + 21m) \mod k_2$$

$$\pi(s) = 2(t + ck_1/2 + 1) - m$$

end for

Decoding



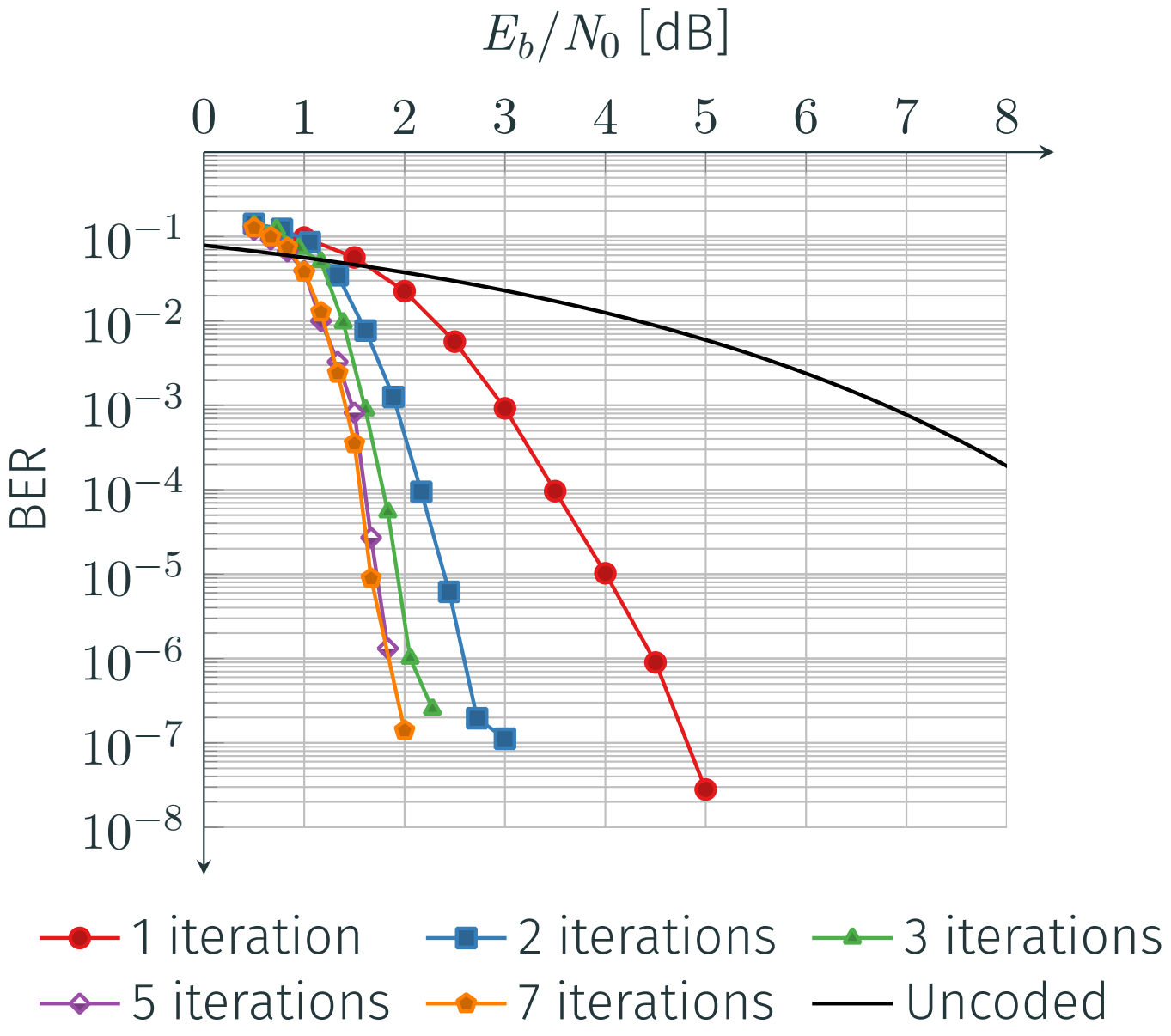
- BCJR (in log domain) on upper and lower code
- puncturing is applied at reception $\hat{r}[i] = r[i] \cdot p[i]$ so that

$$g_l(\mathbf{y}_l) \propto \exp\left(-\frac{\|\mathcal{L}(\mathbf{y}_l)^2\|}{2\sigma_w^2}\right) = \text{const}$$

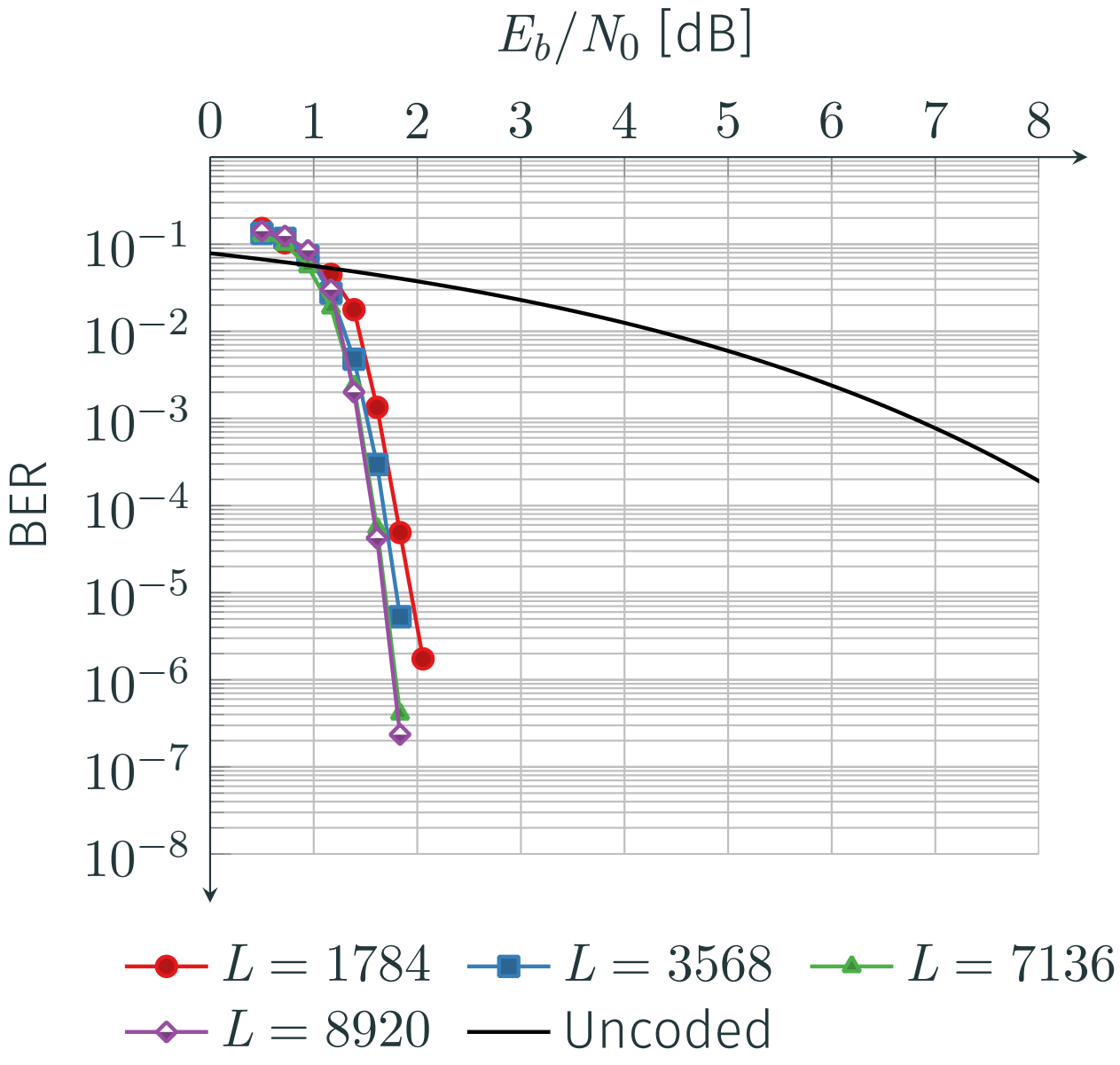
Example: defining a code in C

```
// **messages is initialized to log(0.5)
int *decoded = NULL;
for (int i = 0; i < iterations; i++) {

    // run BCJR on upper code
    convcode_extrinsic(streams[0], lengths[0],
                       &messages, code.upper_code,
                       noise_variance, 0);
    // apply interleaver
    message_interleave(&messages, code);

    // run BCJR on lower code
    // save extrinsic messages in **messages
    decoded = convcode_extrinsic(streams[1], lengths[1],
                       &messages, code.lower_code,
                       noise_variance,
                       i == (iterations - 1));
    // deinterleave
    message_deinterleave(&messages, code);
}
```

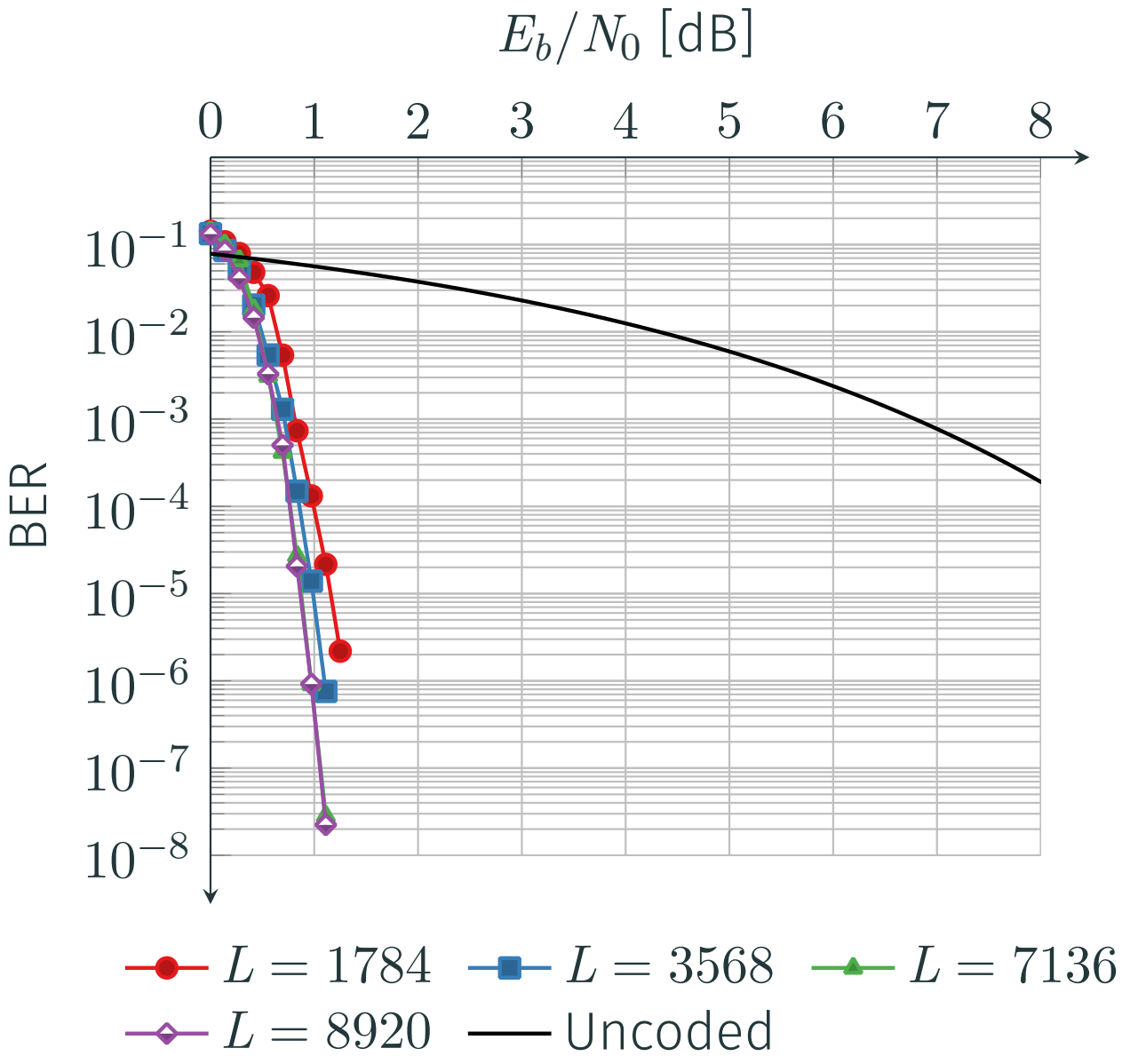
Number of MP iterations



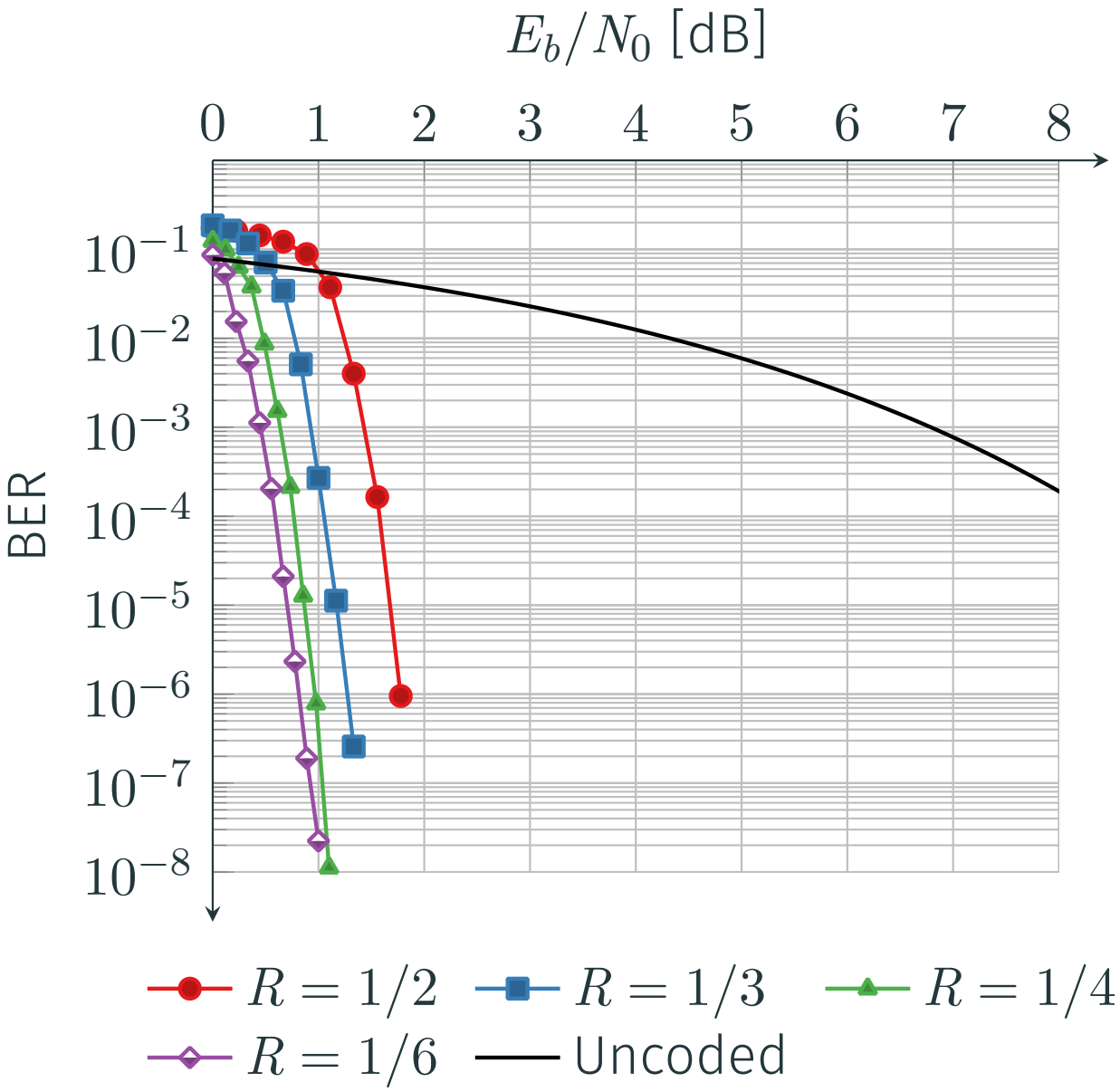
Different packet sizes: $R = 1/2$



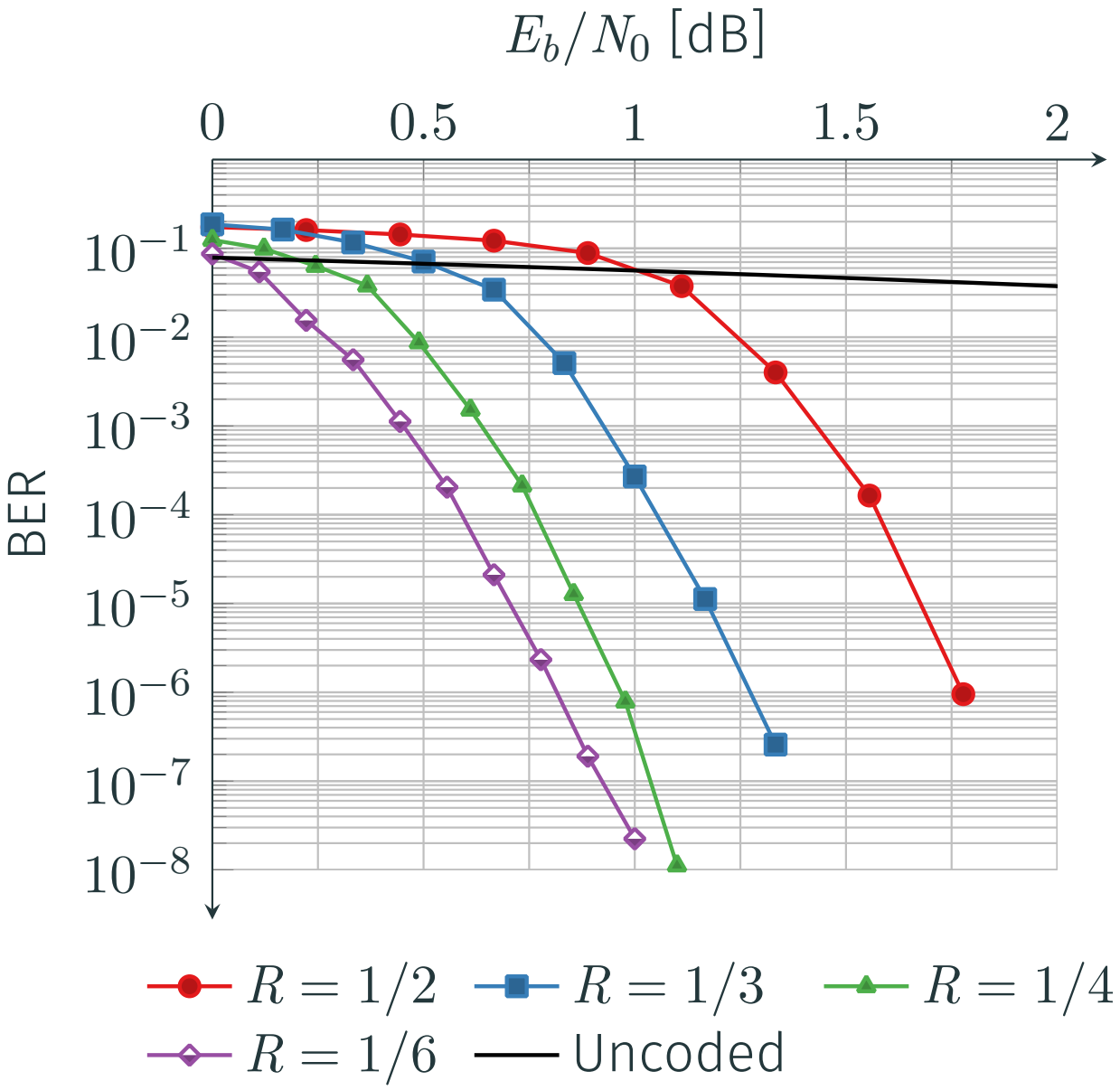
Different packet sizes: $R = 1/4$



Codes comparison: $L = 8920$



Codes comparison: a closer look



Source code for simulator, scripts, plotting and presentation available at

http://github.com/geeanlooca/deepspace-turbo

Thank you!