# On the Complexity of Unary Error Correction Codes for the Near-Capacity Transmission of Symbol Values from an Infinite Set


Wenbo Zhang, Robert G. Maunder, Lajos Hanzo

Presented by

**Rob Maunder**

Electronics and Computer Science,

University of Southampton, SO17 1BJ, UK.

Email:{wz4g11,rm,lh}@ecs.soton.ac.uk

http://cspc.ecs.soton.ac.uk

# Outline

- ❑ Motivation and SSCC benchmarker
- ❑ Proposed JSCC scheme using UEC code
- ❑ Near-capacity analysis
- ❑ Error ratio performance
- ❑ Conclusions and future work

* Separate Source and Channel Coding (SSCC)
* Joint Source and Channel Coding (JSCC)
* Unary Error Correction (UEC) code

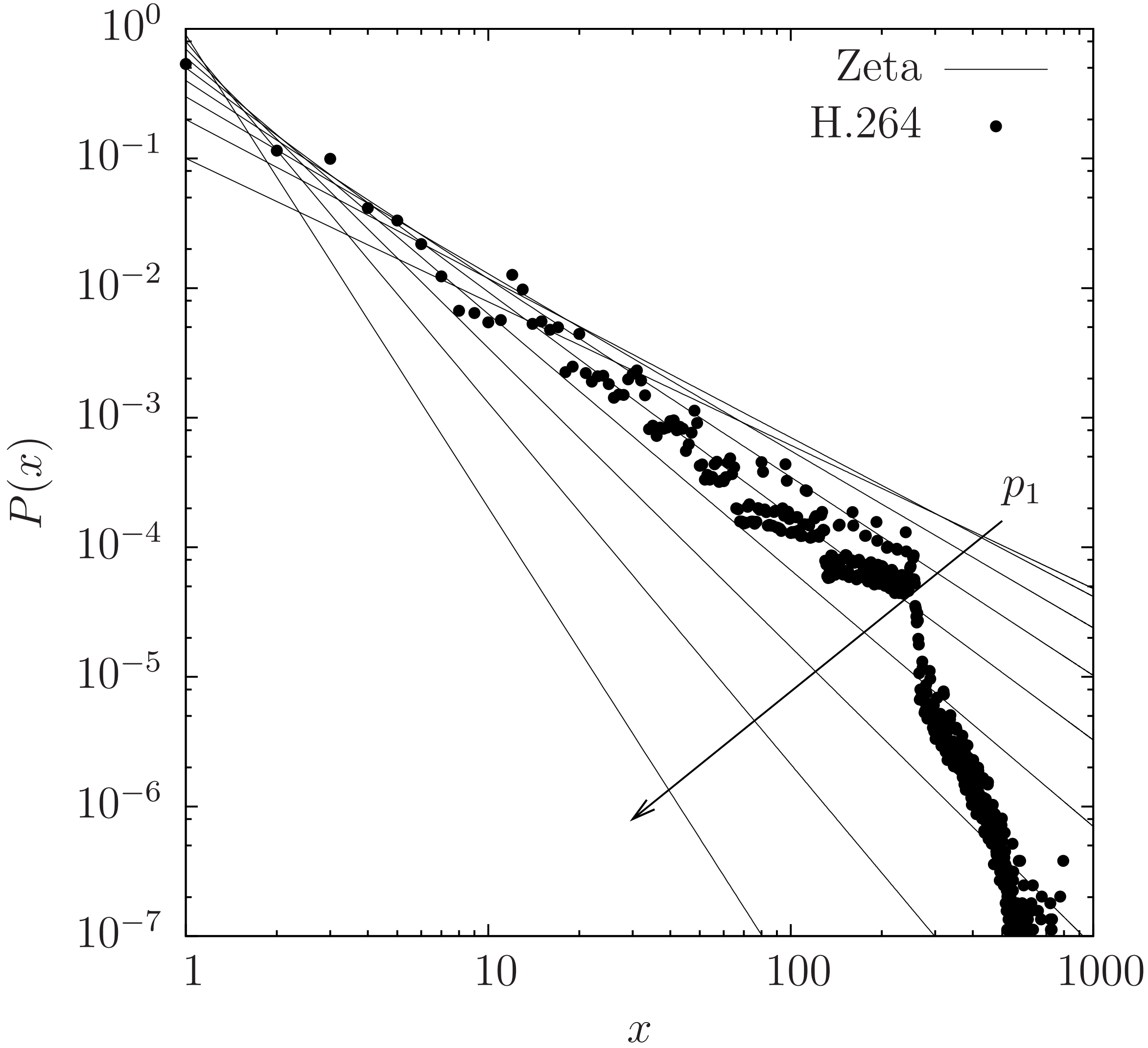
# Background

| | Finite symbol set<br>e.g. $\{a, b, c, \ldots, z\}$ | Infinite symbol set<br>e.g. $\mathbb{N}_1 = \{1, 2, 3, \ldots, \infty\}$ |
|---|---|---|
| Separate Source and Channel Coding (SSCC) | • Huffman code<br>• Shannon-Fano code | • Unary code<br>• Elias Gamma code |
| Joint Source and Channel Coding (JSCC) | • Variable Length Error Correction (VLEC) code | • **Unary Error Correction (UEC) code** |

When decoding symbol values selected from an infinite set:

- existing SSCC schemes have significant capacity loss;
- existing JSCC schemes have infinite complexity.

# Symbol values from an infinite set




Zeta distribution

$P(x) = \frac{x^{-s}}{\zeta(s)}$,

$\zeta(s) = \sum_{x \in \mathbb{N}_1} x^{-s}$,

$s > 1$,

$p_1 = 1/\zeta(s)$.

Symbol entropy

$H_X = \sum_{x \in \mathbb{N}_1} P(x) \cdot \log_2(1/P(x))$.

Here, $p_1 \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$

| $x_i$ | $P(x_i)$ | | | $\mathbf{y}_i$ |
|---|---|---|---|---|
| | $p_1 = 0.7$ | $p_1 = 0.8$ | $p_1 = 0.9$ | EG |
| 1 | 0.7000 | 0.8000 | 0.9000 | 1 |
| 2 | 0.1414 | 0.1158 | 0.0717 | 010 |
| 3 | 0.0555 | 0.0374 | 0.0163 | 011 |
| 4 | 0.0286 | 0.0168 | 0.0057 | 00100 |
| 5 | 0.0171 | 0.0090 | 0.0025 | 00101 |
| 6 | 0.0112 | 0.0054 | 0.0013 | 00110 |
| 7 | 0.0079 | 0.0035 | 0.0007 | 00111 |
| 8 | 0.0058 | 0.0024 | 0.0004 | 0001000 |
| 9 | 0.0044 | 0.0017 | 0.0003 | 0001001 |
| 10 | 0.0034 | 0.0013 | 0.0002 | 0001010 |

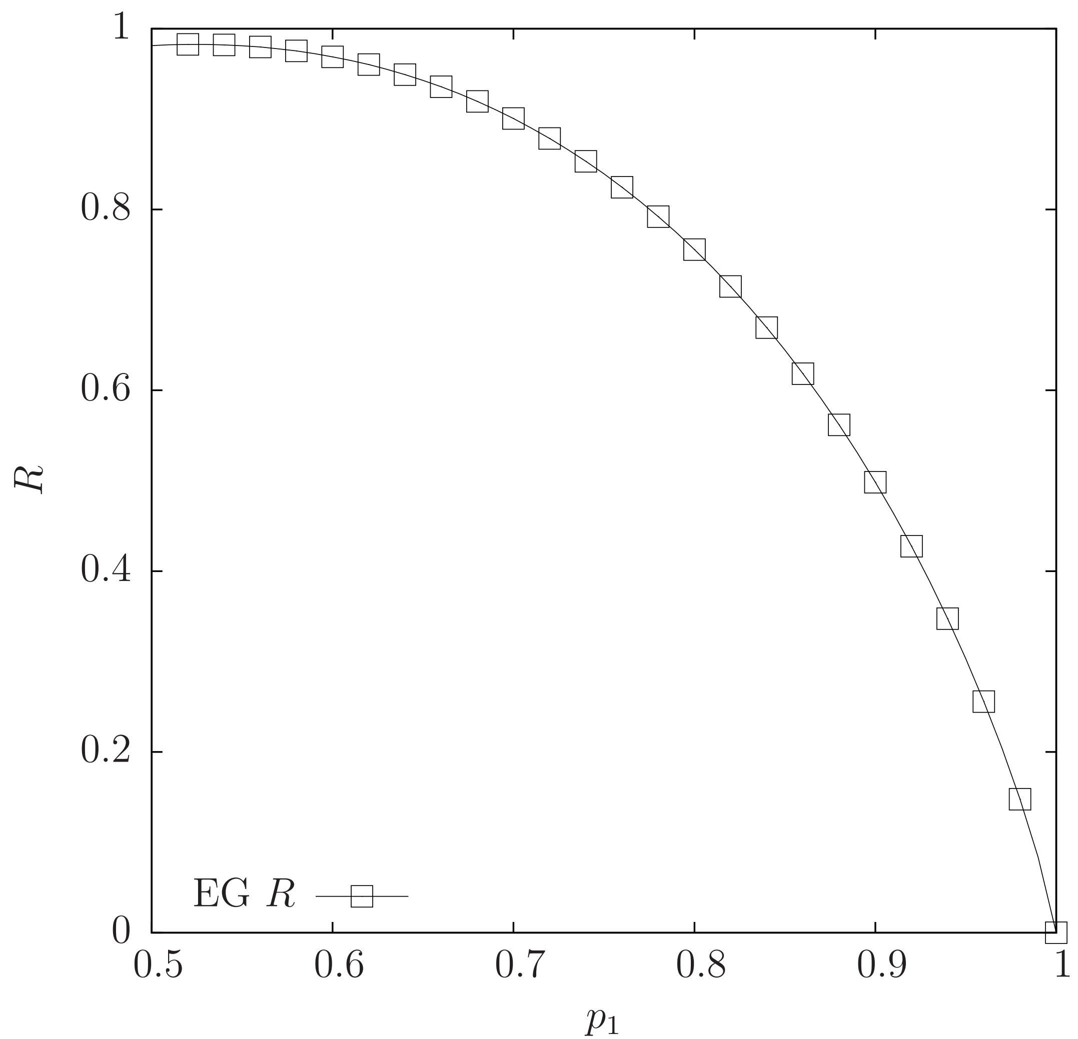
Table 1: The first ten codewords of Elias Gamma (EG) code.

# Elias Gamma code

Average codeword length

$l = \sum_{x \in \mathbb{N}_1} P(x) \left(2\lfloor \log_2(x) \rfloor + 1\right)$.

# EG coding rate



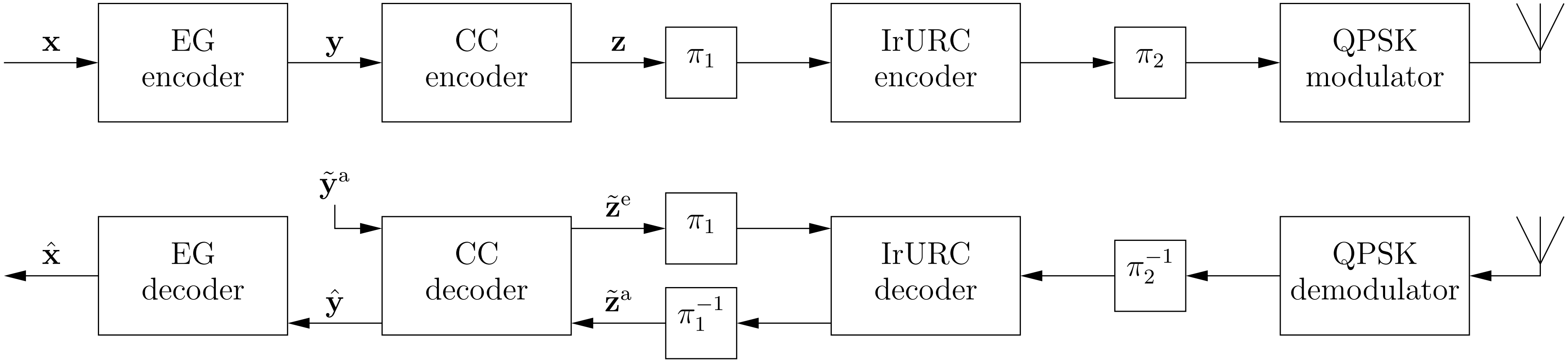

Coding rate of EG code for zeta distribution.

Coding rate

$$R = \frac{H_X}{l}.$$

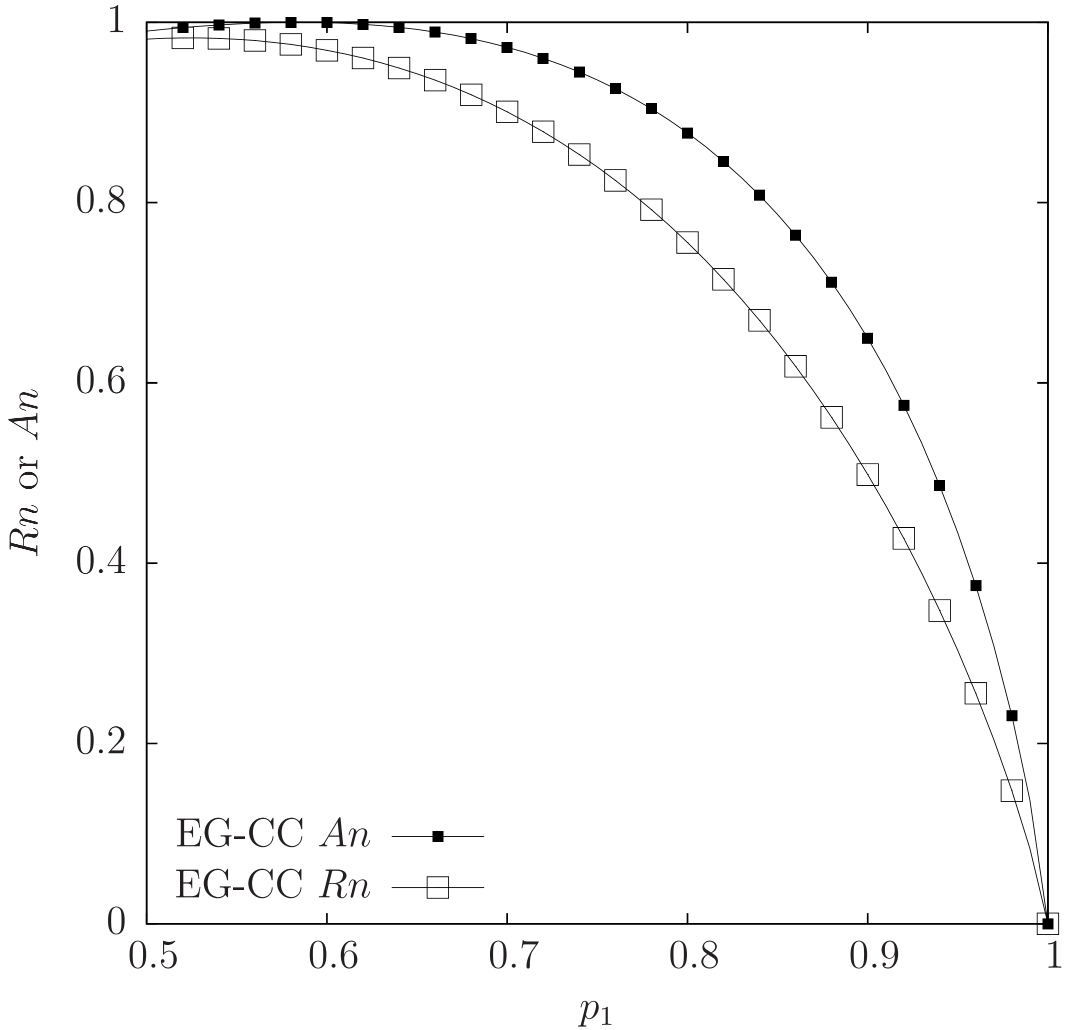
Region above curve represents residual redundancy

# EG-CC SSCC benchmarker




- ⇨ EG code is identical to $k = 0$ Exponential-Golomb code.
- ⇨ Convolutional Code (CC) with $n = 2$ encoded bits, $4$ states and recursive generator polynomial.
- ⇨ Irregular Unity Rate Code (IrURC), whose components have $n = 1$ encoded bit, 2, 4 or 8 states and recursive generator polynomial.
- ⇨ Quaternary Phase Shift Keying (QPSK) with Gray mapping and puncturing.

# Significant EG-CC capacity loss



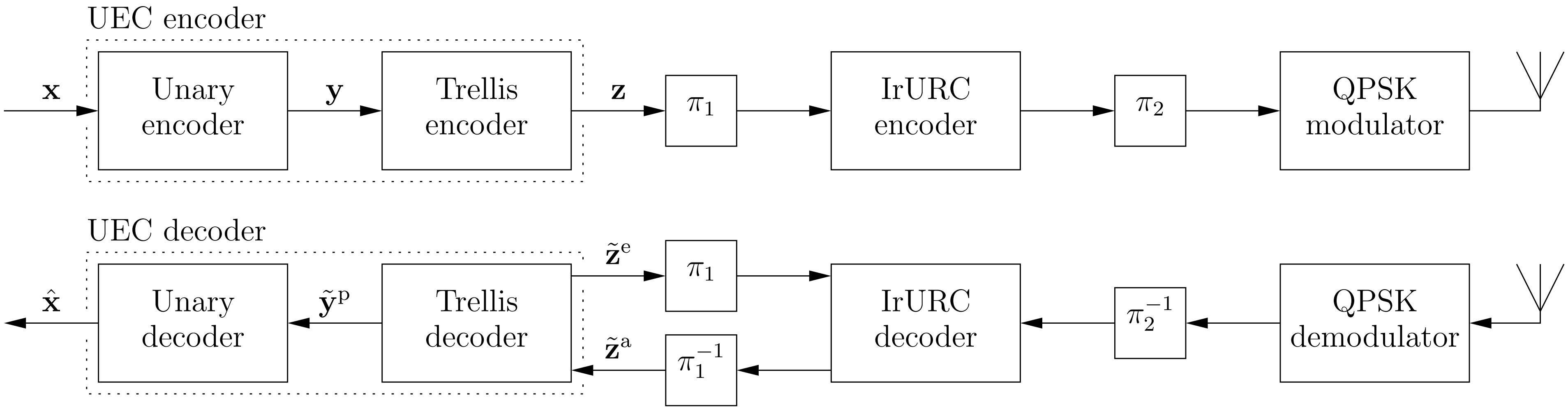

$Rn$ and $An$ of EG-CC scheme, for zeta distribution.

$A$ is area beneath EXtrinsic Information Transfer (EXIT) curve of EG-CC decoder

Region above $An$ curve represents residual redundancy exploited for error correction

Region between curves represents unexploited residual redundancy, giving capacity loss

# Proposed JSCC UEC scheme




⇨ Replace EG code with a unary code.

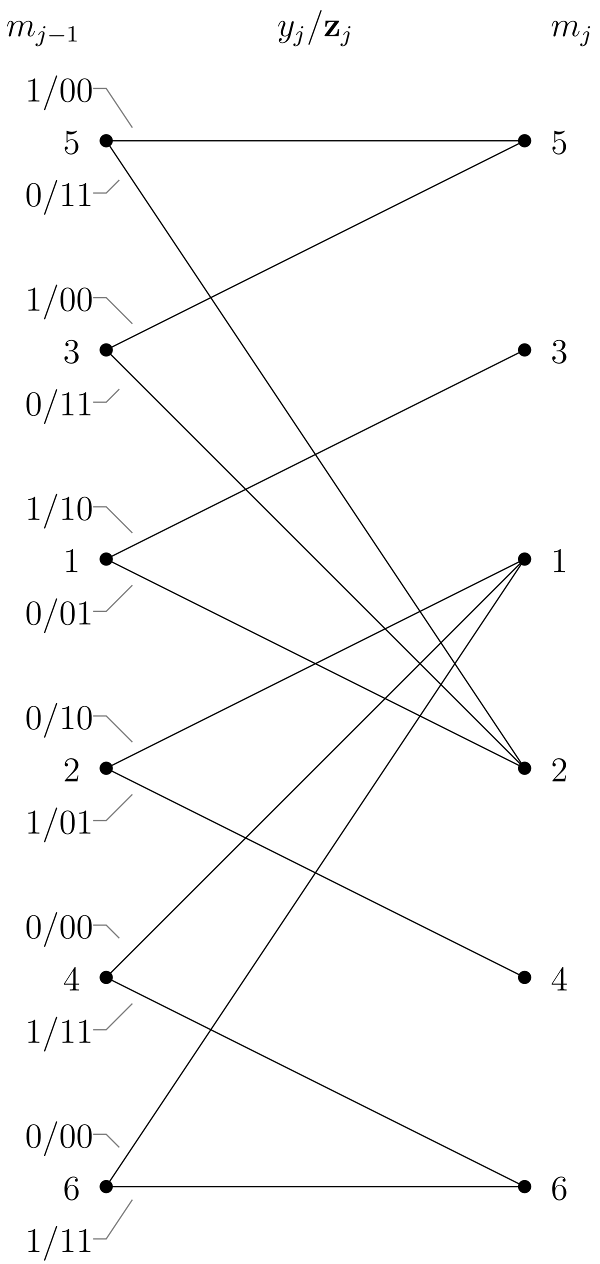
⇨ Replace CC code with novel trellis code, having $n = 2$ encoded bits and $r$ states.

| $x_i$ | $P(x_i)$ | | | $\mathbf{y}_i$ | |
|---|---|---|---|---|---|
| | $p_1 = 0.7$ | $p_1 = 0.8$ | $p_1 = 0.9$ | Unary | EG |
| 1 | 0.7000 | 0.8000 | 0.9000 | 0 | 1 |
| 2 | 0.1414 | 0.1158 | 0.0717 | 10 | 010 |
| 3 | 0.0555 | 0.0374 | 0.0163 | 110 | 011 |
| 4 | 0.0286 | 0.0168 | 0.0057 | 1110 | 00100 |
| 5 | 0.0171 | 0.0090 | 0.0025 | 11110 | 00101 |
| 6 | 0.0112 | 0.0054 | 0.0013 | 111110 | 00110 |
| 7 | 0.0079 | 0.0035 | 0.0007 | 1111110 | 00111 |
| 8 | 0.0058 | 0.0024 | 0.0004 | 11111110 | 0001000 |
| 9 | 0.0044 | 0.0017 | 0.0003 | 111111110 | 0001001 |
| 10 | 0.0034 | 0.0013 | 0.0002 | 1111111110 | 0001010 |

Table 2: The first ten codewords of unary and EG codes.

# Unary code

Average codeword length $l = \sum_{x \in \mathbb{N}_1} P(x)x$.




# Trellis code

Here, the trellis has $r = 6$ states.

Encoding begins in state $m_0 = 1$.
e.g. for symbols $\mathbf{x} = [4, 1, 2, 1, 3, 1, 1, 1, 2, 2]$,

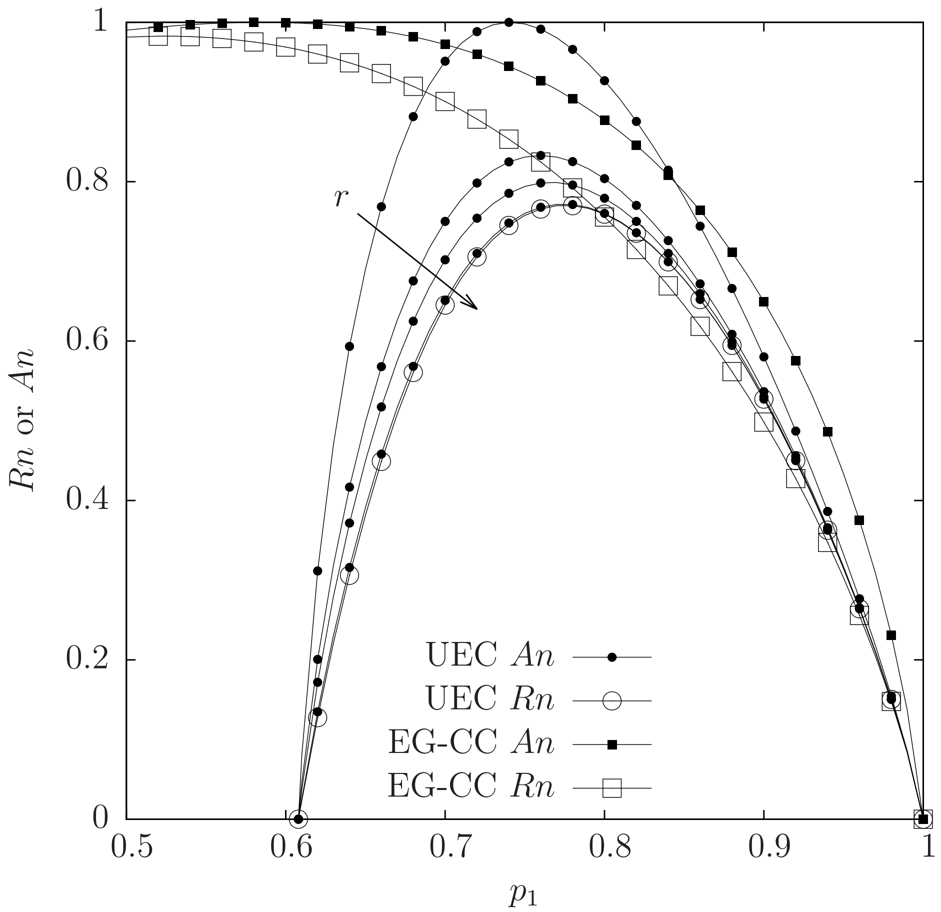
⇨ $\mathbf{y} = [11100100110000101 0]$.

⇨ $\mathbf{m} = [1, 3, 5, 5, 2, 1, 3, 2, 1, 3, 5, 2, 1, 2, 1, 3, 2, 4, 1]$.

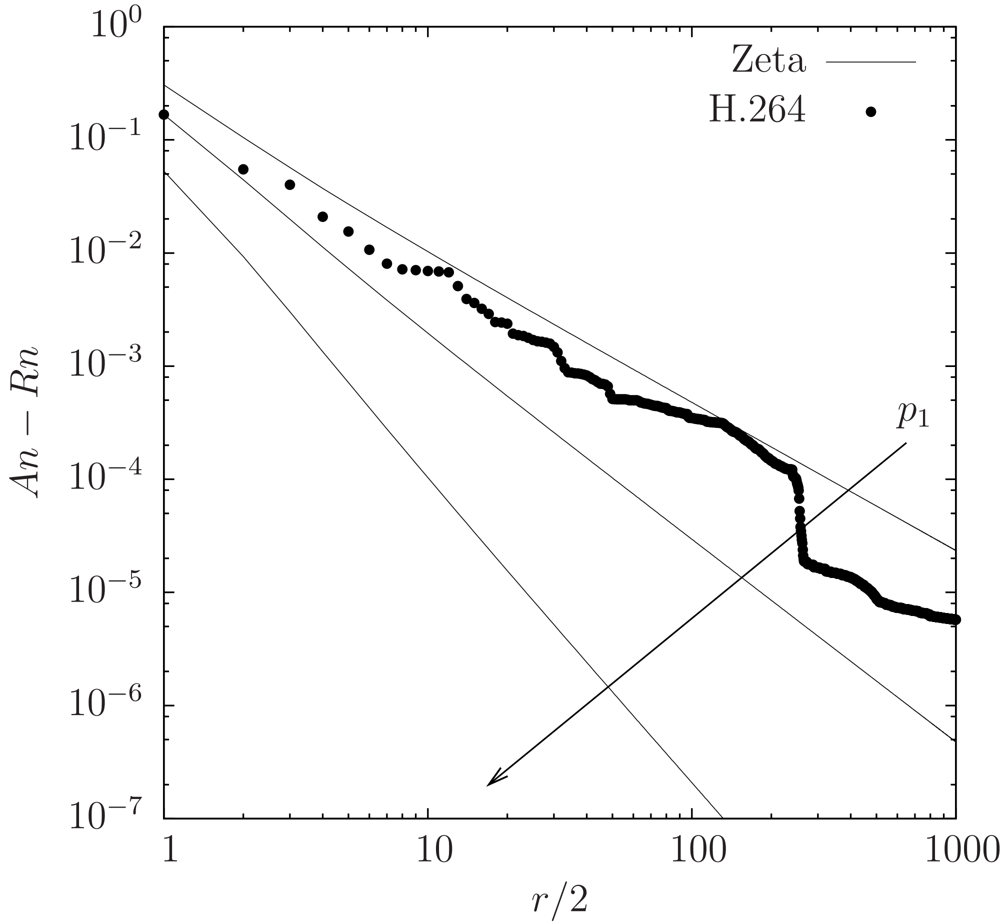
⇨ $\mathbf{z} = [100000111010111010001110011001110100]$.

Each transition occurs with a different probability, which is exploited during soft-in soft-out decoding.

# Vanishing UEC capacity loss

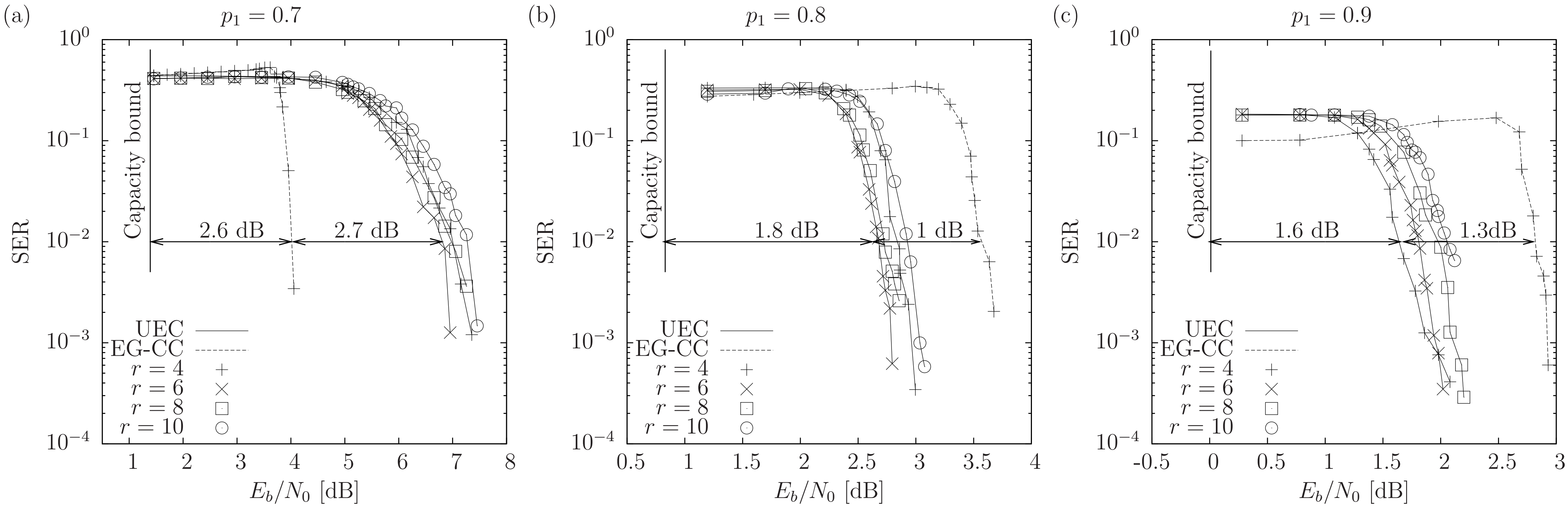



$Rn$ and $An$ of EG-CC scheme and UEC scheme having $r \in \{2, 4, 6, 30\}$ states, for zeta distribution.




Capacity loss in UEC scheme, for zeta distribution having $p_1 \in \{0.7, 0.8, 0.9\}$.

# Symbol Error Ratio (SER) performance




SER performance of EG-CC and UEC schemes, for zeta distribution having $p_1 \in \{0.7, 0.8, 0.9\}$. Uncorrelated narrowband Rayleigh fading channel. $10^4$ symbols per frame and up to $10^4$ Add-Compare-Select (ACS) operations per symbol.

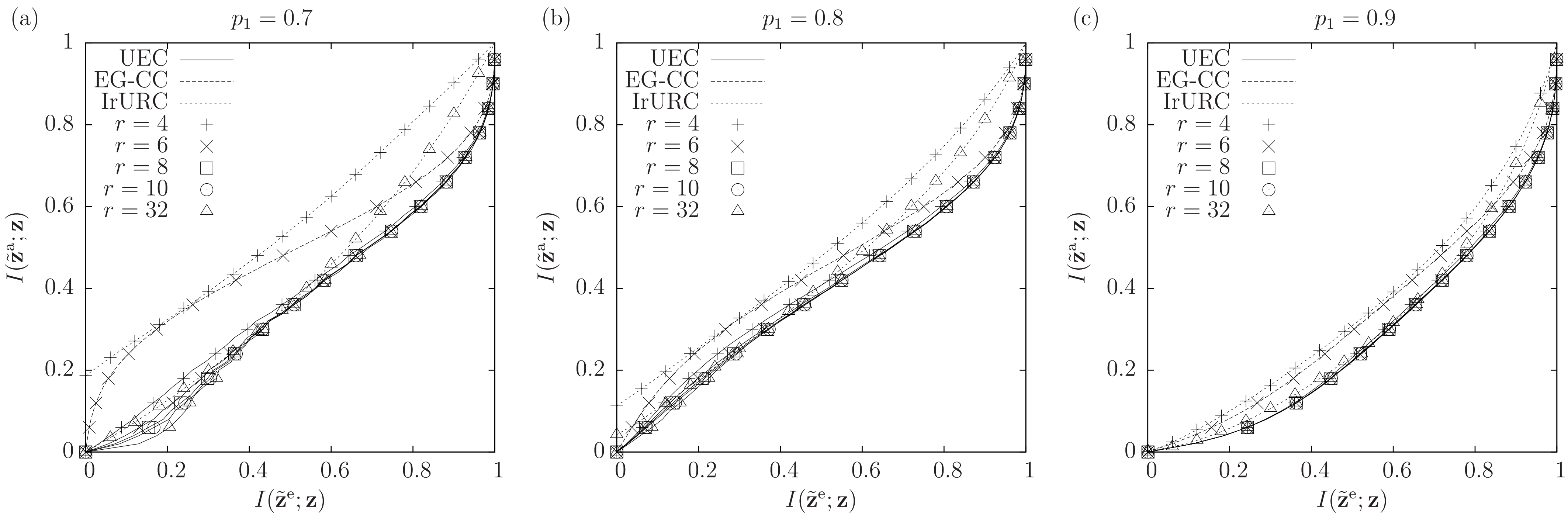
# Conclusions and future work

- SSCC benchmarker suffers from significant capacity loss.
- All previous JSCC schemes have infinite complexity when decoding symbols selected from infinite sets.
- Proposed JSCC UEC scheme has only moderate complexity and its capacity loss asymptotically approaches zero as $r$ increases.
- As much as 1.3 dB gain within 1.6 dB of capacity bound, without any increase in transmission energy, duration, bandwidth or decoding complexity.
- Future work will develop a universal error correction code, which can be applied for all monotonic symbol value distributions.

# Thank you!

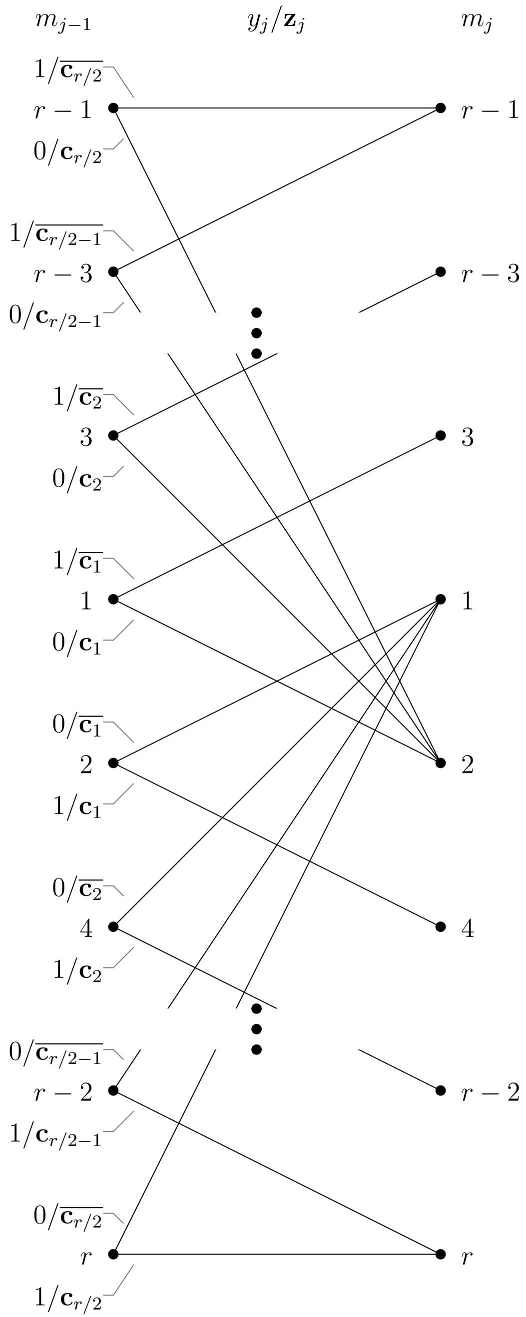
If you liked this, you may also like...

R. G. Maunder, W. Zhang, T. Wang and L. Hanzo, A Unary Error Correction Code for the Near-Capacity Joint Source and Channel Coding of Symbol Values from an Infinite Set, *IEEE Transactions on Communications (In Press)*, available: `http://eprints.soton.ac.uk/341736/`.



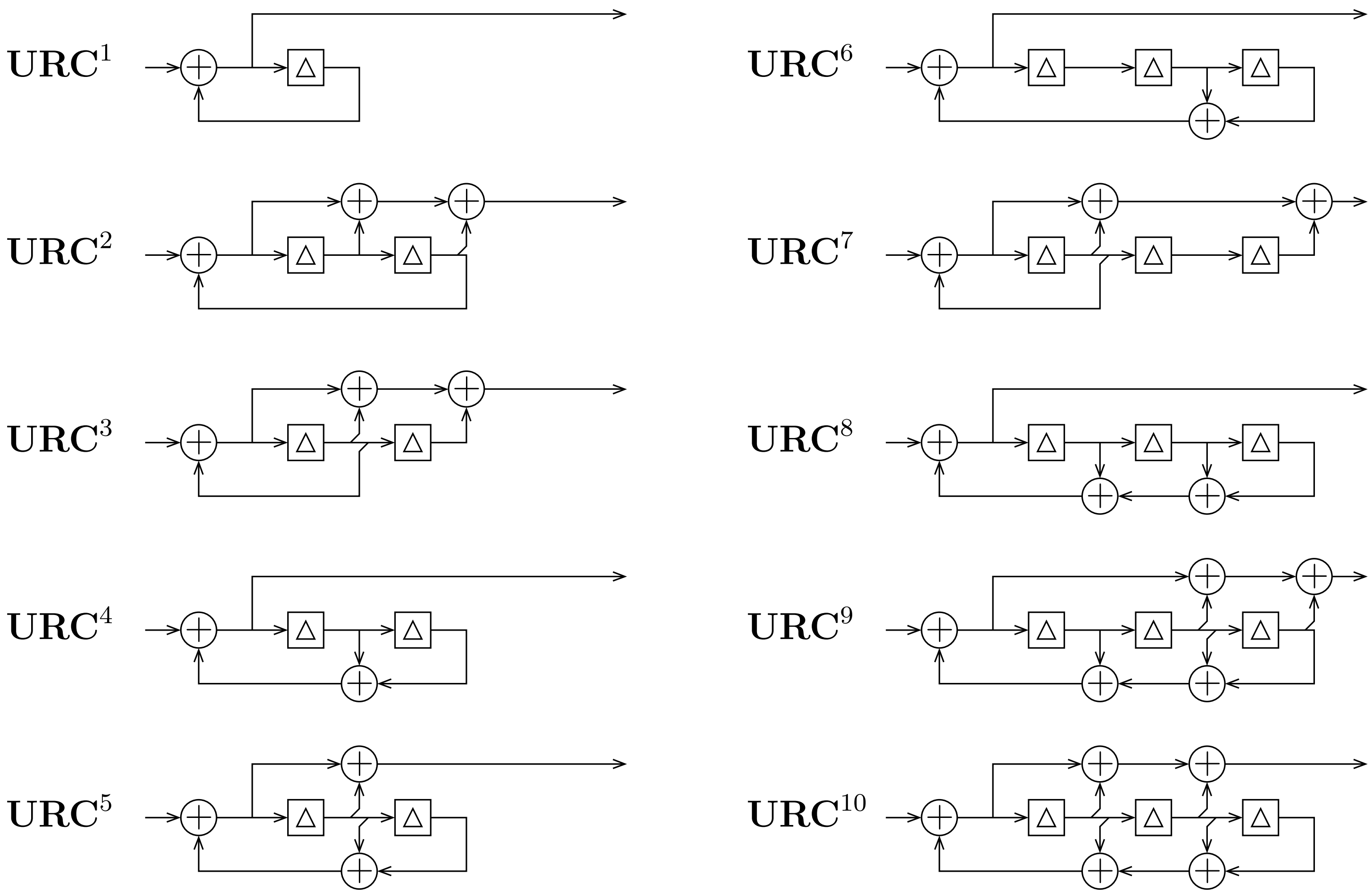

Inverted EXIT curves for the UEC decoder having $r \in \{4, 6, 8, 10, 32\}$ states and EG-CC decoder having $r = 4$ states, where $p_1 \in \{0.7, 0.8, 0.9\}$. Corresponding EXIT curves are provided for the IrURC schemes at the lowest $E_\mathrm{b}/N_0$ values that facilitates the creation of an open tunnel with the EXIT curves of the $r = 32$-state UEC and the $r = 4$-state EG-CC.

# Generalized UEC trellis



$$P(m,m') = \begin{cases} \frac{1}{2l}\left[1 - \sum_{x=1}^{\lceil \frac{m'}{2} \rceil} P(x)\right] & \text{if } m' \in \{1,2,\ldots,r-2\}, m = m'+2 \\ \frac{1}{2l} P(x)\Big|_{x=\lceil \frac{m'}{2} \rceil} & \text{if } m' \in \{1,2,\ldots,r-2\}, m = 1+\mathrm{odd}(m') \\ \frac{1}{2l}\left[1 - \sum_{x=1}^{\frac{r}{2}-1} P(x)\right] & \text{if } m' \in \{r-1,r\}, m = 1+\mathrm{odd}(m') \\ \frac{1}{2l}\left[l - \frac{r}{2} - \sum_{x=1}^{\frac{r}{2}-1} P(x)\left(x-\frac{r}{2}\right)\right] & \text{if } m' \in \{r-1,r\}, m = m' \\ 0 & \text{otherwise} \end{cases}$$

# IrURC component codes



| $p_1$ | IrURC component code fractions $\alpha$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $r = 2$ | $r = 4$ | | | | $r = 8$ | | | | |
| | (2,3) | (7,5) | (7,6) | (4,7) | (6,7) | (8,B) | (D,C) | (8,F) | (B,F) | (E,F) |
| 0.7 | 0 | 0 | 0.44 | 0 | 0.44 | 0 | 0.10 | 0 | 0.02 | 0 |
| | 0.35 | 0 | 0 | 0.18 | 0.17 | 0.05 | 0 | 0.25 | 0 | 0 |
| 0.8 | 0.18 | 0 | 0.71 | 0.10 | 0.01 | 0 | 0 | 0 | 0 | 0 |
| | 0.30 | 0 | 0.33 | 0.27 | 0.10 | 0 | 0 | 0 | 0 | 0 |
| 0.9 | 0 | 0 | 0.33 | 0 | 0 | 0.09 | 0.58 | 0 | 0 | 0 |
| | 0 | 0 | 0.85 | 0 | 0 | 0.02 | 0.13 | 0 | 0 | 0 |

Table 3: The fraction of the IrURC input bit sequence that is encoded by each component code.

| Decoder | $r$ | max* | add | ACS |
|---|---|---|---|---|
| $n = 2$-bit CC Viterbi decoder $\hat{\mathbf{y}}$ | 4 | 2 | 8 | 18 |
| $n = 2$-bit CC BCJR decoder $\tilde{\mathbf{z}}^{\mathrm{e}}$ | 4 | 10 | 22 | 72 |
| $n = 2$-bit Trellis BCJR decoder $\tilde{\mathbf{y}}^{\mathrm{p}}$ | 4 | 7 | 20 | 55 |
| | 6 | 11 | 30.5 | 85 |
| | 8 | 15 | 40.5 | 115.5 |
| $n = 2$-bit Trellis BCJR decoder $\tilde{\mathbf{z}}^{\mathrm{e}}$ | 4 | 10 | 22 | 72 |
| | 6 | 16 | 32 | 112 |
| | 8 | 22 | 42 | 152 |
| URC BCJR decoder | 2 | 6 | 19 | 49 |
| | 4 | 14 | 37 | 107 |
| | 8 | 30 | 73 | 223 |

# Scheme parametrizations

| $p_1$ | Scheme | $r$ | $R_o$ | $A_o$ | $R_i$ | $\eta$ | $E_b/N_0$ $[dB]$ for $C=\eta$ | $E_b/N_0$ $[dB]$ for $A_i = A_o$ | $E_b/N_0$ $[dB]$ for open tunnel |
|---|---|---|---|---|---|---|---|---|---|
| 0.7 | EG-CC | 4 | 0.4503 | 0.4861 | 1 | 0.9006 | 1.39 | 2.03 | 3.5 |
| | UEC | 4 | 0.3226 | 0.3751 | 1.3958 | | | 2.70 | 3.8 |
| | | 6 | | 0.3510 | | | | 2.09 | 3.7 |
| | | 8 | | 0.3412 | | | | 1.85 | 3.7 |
| | | 10 | | 0.3361 | | | | 1.72 | 3.6 |
| | | 32 | | 0.3253 | | | | 1.46 | 3.4 |
| 0.8 | EG-CC | 4 | 0.3779 | 0.4387 | 1.0048 | 0.7594 | 0.83 | 1.96 | 3.1 |
| | UEC | 4 | 0.3797 | 0.4019 | 1 | | | 1.24 | 2.4 |
| | | 6 | | 0.3896 | | | | 1.01 | 2.0 |
| | | 8 | | 0.3853 | | | | 0.92 | 1.8 |
| | | 10 | | 0.3833 | | | | 0.90 | 1.8 |
| | | 32 | | 0.3801 | | | | 0.84 | 1.8 |
| 0.9 | EG-CC | 4 | 0.2492 | 0.3247 | 1.0578 | 0.5272 | 0.01 | 1.72 | 2.2 |
| | UEC | 4 | 0.2636 | 0.2682 | 1 | | | 0.11 | 0.9 |
| | | 6 | | 0.2651 | | | | 0.04 | 0.9 |
| | | 8 | | 0.2642 | | | | 0.02 | 0.8 |
| | | 10 | | 0.2639 | | | | 0.01 | 0.8 |
| | | 32 | | 0.2636 | | | | 0.01 | 0.7 |

Outer coding rate $R_o$, inner coding rate $R_i$ and total throughput $\eta$ for two schemes with different values of $p_1$ and $r$. Three categories of $E_b/N_0$ where $C = \eta$ and $A_i = A_o$ in theory, and where tunnel is open in simulation, respectively.