UNIVERSIDADE FEDERAL DE SANTA CATARINA

CENTRO TECNOLÓGICO

DEPARTAMENTO DE ENGENHARIA ELÉTRICA E ELETRÔNICA

GUSTAVO SIMAS DA SILVA

TRABALHO 1 – INTRODUÇÃO À CODIFICAÇÃO

FLORIANÓPOLIS

SETEMBRO, 2019

# Introdução

O atual trabalho tem como objetivo realizar uma simulação computacional fazendo uso do programa MatLab, de maneira a se avaliar a probabilidade de erro de palavra-código. São implementados 3 códigos de bloco:

1. Código de repetição, com taxa R = 1/7;
2. Código de Hamming, com taxa R = 4/7;
3. Código de transmissão, com taxa R = 7/7.

O canal adota é o BSC (Binary Symmetric Channel – Canal Binário Simétrico), com probabilidade de transição **p**. São realizadas 4 simulações, uma para cada valor de p.

1. p = 0,05;
2. p = 0,1;
3. p = 0,2;
4. p = 0,3.

São geradas sequências de bits de informação (com iguais probabilidades para **0** e para **1**) de tamanho **L** blocos, de **k** bits cada bloco. Utiliza-se de um tamanho **L > 1000**, a princípio, se tendo maior precisão à medida que L aumenta, no entanto havendo limitações por conta do tempo de processamento do programa.

Um bloco de **k** bits, dadas tais características, é obtido com a operação ***u = round(rand(1,k))***. Isto pois o comando *rand(a,b)* gera uma matriz de ***a*** linhas e ***b*** colunas, com número aleatórios entre 0 e 1; enquanto o comando *round* efetua o arredondamento de cada elemento da matriz gerada para o decimal inteiro mais próximo, resultando, portanto, em ***a***blocos (vetores) de ***b*** bits cada. Podemos obter a palavra-código **v = uG** pela matriz geradora:

As palavras-código são salvas na memória, para futura comparação com as palavras-código decodificadas. O programa desenvolvido gera, também, uma sequência de bits de tamanho igual a **L** blocos de 7 bits cada, para representar o ruído binário. Para tal sequência, a probabilidade de o bit ser **1** é **p**, quanto a probabilidade de o bit ser **0** é **1-p**.

Cada bloco composto (padrão de erro) é gerado por meio do comando ***e = round(rand(1,7) – 0.5 + p)***. Isto faz com que o vetor padrão de erro seja dependente da probabilidade de o bit ser **1**, pois posteriormente será somado à palavra-código. Assim, para um valor baixo de **p**, o vetor de erro **e** terá menos elementos iguais a **1** (menor ruído); enquanto que, para um valor alto de **p**, o padrão de erro será mais ruidoso (mais elementos iguais a **1**).

A decodificação do código de repetição é feita por lógica majoritária, ou seja, se o peso de Hamming (número de **1’s**) do vetor recebido for maior que o número de **0’s**, então decide-se pela palavra-código decodificada **v’ = (1111111)**. Caso contrário, **v’ = (0000000)**.

Para o código Hamming, utiliza-se decodificação exaustiva, ou seja, calcula-se a distância de Hamming entre o vetor recebido **r = v + e** e todas as palavras-código do código. Assim, é escolhida um **v’** mais próximo de **r**. No caso de empate, é escolhida qualquer uma delas, não havendo diferença significativa a longo prazo.

A decodificação do código de transmissão (taxa 7/7) não existe propriamente. Apenas aceita-se o que é recebido, sendo **v’ = r**.

Depois da decodificação, o programa computa o número de palavras-código decodificadas erroneamente, sendo isto feito ao se comparar a palavra-código transmitida **v** (salva na memória) com a decodificada **v’**. A taxa de erro é obtida dividindo-se o número de palavras-código decodificadas erroneamente 1 pelo número total de palavras-código transmitidas (**L**).

Por fim, são apresentados os resultados na forma de gráficos **Perro x p**, para cada código simulado, tendo-se a probabilidade de erro simulada e teórica.

Para o **código de repetição,** a probabilidade de erro teórica é dada por:

Para o **código de Hamming,** a probabilidade de erro teórica é dada por:

# Resultados