ELE-32 Introdução a Comunicações

Aula 4- Uma comparação justa

November 14, 2018

1 Introdução

Dada uma potência P, uma taxa de bits R_b e uma modulação conhecida, gastamos uma quantidade média de E_b J por bit transmitido através de um canal. Na ausência de codificação de canal, o valor de E_b é o valor que de fato utilizamos por bit de informação. Assim, o valor de energia por bit de informação é $E_i = E_b$

Entretanto, manter o valor de E_b e introduzir um método de codificação de canal efetivamente aumenta o valor de E_i como demonstra o raciocínio a seguir. Codificamos k bits de informação em n bits transmitidos. Logo, gastamos nE_b de energia para transmitir k bits de informação, isto é, $kE_i = nE_b$. Como $k < n$, $E_i = \frac{nE_b}{k} > E_b$. Utilizando a taxa do codificador $R = \frac{k}{n}$, podemos escrever $E_i = \frac{E_b}{R}$.

É injusto comparar sistemas com valores diferentes de E_i pois poderíamos melhorar o desempenho daquele com menor E_i simplesmente até o ponto em que este valor seja igual para todos os sistemas. Técnica semelhante foi utilizada para comparar modulações diferentes nas aulas de teoria.

Além disso, até o momento as nossas curvas de probabilidade de erro tem tido como argumento o parâmetro p de um canal BSC. Este valor p é a probabilidade de erro de bit. Já vimos na teoria que é possível estimar p em função de $\frac{E_b}{N_0}$. Por exemplo, para a modulação BPSK, $p\left(\frac{E_b}{N_0}\right) = Q\left(\sqrt{\frac{2E_b}{N_0}}\right)$. Obtivemos as estimativas de probabilidades de erro de bit de informação em função de p. Por exemplo, a probabilidade de erro de bit de informação do código de Hamming é uma curva do tipo $P_H(p)$. Assumindo que estamos utilizando uma modulação BPSK, podemos facilmente obter $P_H\left(\frac{E_b}{N_0}\right)$.

Unindo estas duas formas é possível comparar todos os sistemas estudados nos laboratórios anteriores e as variações do algoritmo de Viterbi a serem geradas neste laboratório.

2 Pequenas variações do algoritmo de Viterbi

No laboratório passado foi utilizado como custo associado a cada ramo a distância de Hamming entre o que foi recebido e o que esperaríamos receber caso a transição daquele ramo tivesse acontecido. Esta distância é uma aproximação para o logarítmo da probabilidade de ter acontecido aquela transição. Por exemplo, se recebermos através de um canal BSC com parâmetro p a subsequência 000 e houver um ramo com rótulo de saída 001, os valores de probabilidade seriam:

$$\begin{aligned} P[\mathbf{s} = 001|\mathbf{r} = 000] &\propto (1-p)(1-p)p \\ &= (1-p)^2 p^1 \\ log[P[\mathbf{s} = 001|\mathbf{r} = 000]] &= 2log[1-p] + 1log[p] + K \end{aligned} \tag{1}$$

Como $log[x]$ é uma função monotonicamente crescente, o valor de x que maximiza $f(x)$ também maximizará $log[f(x)]$. Se p for muito pequeno, $log(p) << log(1-p) \approx 0$ (ambos são negativos). O valor que multiplica o termo $log[p]$ é a distância de Hamming entre a subsequência recebida e a esperada. Quanto menor for a distância, menos negativo será $log[P[\mathbf{s}|\mathbf{r}]]$. Além disso, sendo o logaritmo do produto a soma dos logaritmos, esta probabilidade é acumulada ao somarmos o custo do ramo ao custo do estado de partida.

A **primeira variação** possível é utilizar o valor exato de $log[P[\mathbf{s}|\mathbf{r}]]$ no lugar da distância de Hamming. Esta variação deve influenciar na probabilidade de erro final quando o valor de p não é tão pequeno, o que torna o valor de $log(1-p)$ razoavelmente diferente de 0.

O mesmo raciocínio pode ser aplicado quando transmitimos uma sequência binária utilizando uma modulação digital qualquer através de um canal AWGN. Como vimos na teoria, para símbolos equiprováveis, o símbolo mais provável é o mais próximo do recebido. Para chegar nesta conclusão também utilizamos o critério de Máxima a Posteriori, símbolos equiprováveis, e a função logaritmo. No contexto do algoritmo de Viterbi, isto é implementado utilizando como métrica a distância Euclidiana entre o valor recebido e os valores possivelmente transmitidos em um dado instante. Neste caso os valores recebidos pertencem a $\Re^n$, onde n é o número de bits transmitidos por seção de treliça. O objetivo é agora obter o caminho com **menor** distância Euclidiana.

O que de fato está sendo feito é postergar o momento da decisão: no caso do canal BSC tomamos a decisão sobre o bit transmitido antes da decodificação; no caso do canal AWGN nem precisamos tomar esta decisão. Utilizar como métrica a distância Euclidiana é a **segunda variação** que veremos.

Há outras variações que não serão vistas neste laboratório, e.g., algoritmo BCJR.

3 Atividade

1. Obtenha analiticamente a curva $p(E_i/N_0)$
2. Converta as curvas dos laboratórios anteriores de $P_e(p)$ para $P_e(E_i/N_0)$ utilizando a função obtida no item anterior.
3. Implemente as duas variações do algoritmo de Viterbi
4. Obtenha as curvas de probabilidade de erro de bit de informação para as duas variações. Para a primeira utilize o canal BSC. Para a segunda será necessário utilizar a modulação BPSK através de um canal Gaussiano com parâmetro $\frac{E_b}{N_0}$ apropriadamente escolhido para corresponder a mesma faixa de valores de p utilizados no caso BSC.
5. Compare todos os sistemas do semestre utilizando como referência o mesmo valor de $\frac{E_i}{N_0}$

4 Perguntas a serem respondidas no relatório

Além das perguntas provenientes do relatório anterior, compare todos os sistemas analisados nos laboratórios anteriores utilizando como referência o valor de $\left(\frac{E_b}{N_0}\right)$. Nominalmente os sistemas são:

- Sistema não codificado
- Código de Hamming
- Código gerado por você
- Códigos cíclicos
- Codificação convolucional com métricas:
  - Distância de Hamming
  - Probabilidades do canal BSC associado

– Distância Euclidiana.

Compare todos com a capacidade do canal Gaussiano estabelecendo o valor de $\left(\frac{E_b}{N_0}\right)$ que teoricamente permitiria a transmissão com probabilidade de erro tão baixa quanto se queira e o valor de $\left(\frac{E_b}{N_0}\right)$ de fato necessário para transmitir com os sistemas acima com probabilidade de erro de 10^{-4}.

5 Referências

- https://en.wikipedia.org/wiki/Forward_error_correction
- https://en.wikipedia.org/wiki/Convolutional_code
- http://www.ele.ita.br/~manish/eet49/Documents/EET49-20120530-Notas%20de%20aula%2010.pdf
- http://www.ele.ita.br/~manish/eet49/Documents/EET49-20120806-Notas%20de%20aula%2012.pdf
- https://en.wikipedia.org/wiki/Turbo_code
- https://en.wikipedia.org/wiki/BCJR_algorithm