1 Introdução

Esse trabalho visa demonstrar as vantagens/desvantagens de várias formas de transmissão de informação. Para isso, vamos simular a transmissão de um sinal usando os seguintes métodos:

1. AM com demodulação síncrona.
2. FM com demodulação síncrona.
3. PCM com pulso SRRC.

A sugestão é que se use para as simulações a linguagem `python`, usando a interface do jupyter notebook/lab, para que se possa visualizar ou até mesmo escutar caso a transmissão seja de um arquivo de áudio. Além disso, com uma instalação básica, é possível usar as bibliotecas de computação numpy e scipy que irão facilitar a execução do trabalho com as funções prontas. Naturalmente, se o aluno desejar usar outras ferramentas, está livre para fazê-lo.

Sugestão

Vale a pena usar um gerenciador de pacotes para o python. Eu sugiro o Mambaforge para instalações e atualizações mais rápidas. Funciona em Linux, Windows ou Mac.

2 Instruções

Durante as simulações, naturalmente não temos como ter sinais *contínuos no tempo*, portanto, usaremos a seguinte estratégia:

- Banda base: os sinais de informação terão taxa de amostragem de 9600 kHz, e
- Banda intermediária: os sinais modulados terão taxa de amostragem de 2 MHz.

Isso permitirá que 'emulemos' as condições de transmissão em frequências mais altas dos sinais de banda base.

Sugestão

Para converter entre banda base e banda intermediária, use a função signal.resample_poly. Verifique se a conversão de ida/volta é consistente. Se a ida/volta não for consistente, verifique se o sinal está criticamente amostrado; se a margem de amostragem for muito baixa, a reamostragem pode introduzir erros.

Importante

O processo de filtragem introduz atrasos a menos que se use um filtro de fase-zero. Se houver atraso entre os sinais transmitidos e demodulados, $|m(t)-\hat{m}(t)|$ será alto não por efeitos de transmissão mas por um problema de medida, portanto utilize um dos seguintes métodos:

- Se usando um filtro IIR (Butterworth, Chebyschev, Bessel, Cauer), use a função signal.filtfilt. Isso irá filtrar o sinal nas duas direções, efetivamente dobrando a ordem do filtro, além de não introduzir atrasos.
- Se usando um filtro FIR simétrico (Remez, LS, Window), use a função signal.convolve com o argumento mode='same'. Isso irá 'centralizar' o resultado da convolução, efetivamente cancelando os atrasos.

2.1 AM síncrono

No AM síncrono, simplesmente multiplicamos o sinal modulante por uma portadora senoidal, enquanto na demodulação multiplicamos o sinal modulado por uma senoide *em fase* com a portadora e em seguida passamos um passa baixas. Lembre-se que $\cos^2(t) = \frac{1}{2}(1+\cos(2t))$, portanto devemos ter um fator multiplicativo igual a 2 para que os sinais sejam numericamente iguais.

(i) Gere um sinal banda-base $m[n]$ entre 0 e 3.5 kHz, com média zero e potência média igual a 1 V^2, e duração de 5 segundos. Para gerar um sinal com essa banda, basta gerar um sinal aleatório gaussiano branco e filtrá-lo com um filtro passa-baixas; fase-zero não é necessário nesse caso. Para a potência média, normalize o sinal pela raiz quadrada de sua potência.

(ii) Reamostre e module $m[n]$ por uma portadora senoidal de frequência 100 kHz e amplitude 1 V, gerando $x_{AM}[t]$. Verifique que a densidade espectral de potência

de $x_{AM}[t]$ está centralizada na frequência da portadora, e que a largura de banda é compatível com o esperado.

(iii) Demodule $x_{AM}[t]$ com uma portadora em fase, filtrando com um passa-baixas de frequência de corte em 4 kHz. Em seguida, reamostre o sinal filtrado de volta para a taxa de banda-base. Verifique que o sinal demodulado é muito próximo do sinal transmitido (média de $(m[n] - \hat{m}[n])^2$ próxima a zero.)

(iv) Repita o processo do item (iii), porém com adição de ruído. Gere um ruído gaussiano de variância unitária $w[t]$, distribuído em banda entre 0 e 500 kHz, e some-o à $x_{AM}[t]$ antes da demodulação, gerando os sinais $\hat{x}[t] = x[t] + \alpha w[t]$, para vários valores de α; Definindo a razão sinal-ruído da mensagem, SNR_m, pela razão entre a potência de $m[n]$ e a potência da diferença $m[n] - \hat{m}[n]$, e a razão sinal-ruído de entrada como a razão entre a potência de $x_{AM}[t]$ e a de $\alpha w[t]$, SNR_i. Grafe os pontos SNR de entrada $\times$ mensagem. O que você observa? Comente.

2.2 FM síncrono

No FM síncrono, o sinal modulado altera a frequência de uma portadora que tem amplitude constante. Na realidade, uma implementação mais fácil do FM envolve um modulador PM, que pode ser implementado por um modulador em quadratura.

(i) Gere um sinal $m[n]$ similar ao do item (i) da subseção anterior.

(ii) Module, usando PM, o sinal $\beta m[n]$ por uma portadora senoidal em 100 kHz, gerando o sinal $x_{PM}[t]$. Garanta que sua simulação comporte vários valores de β. A amplitude de $x_{PM}[t]$ deve ser 1 V.

(iii) Verifique que a densidade espectral de potência de $x_{PM}[t]$ está centralizada na frequência da portadora, e que está compatível com o esperado pela teoria.

(iv) Demodule $x_{PM}[t]$ com um demodulador em quadratura. Para isso, as funções np.arctan2 e np.unwrap podem ser úteis. Qual será a frequência de corte adequada para o filtro de demodulação? Verifique que seu modulador/demodulador PM funciona sem erros para vários valores de β.

(v) Compute $m'[n]$, a integral do sinal $m[n]$. Use seu modulador PM com $\beta m'[n]$, efetivamente criando o modulador FM. Verifique que esse modulador funciona sem erros, e que a densidade espectral de potência de $x_{FM}[t]$ corresponde ao

esperado. Para esse item, as funções np.cumsum, np.diff e np.pad podem ser úteis.

(vi) Similarmente ao item (iv) da subseção AM, adicione um ruído $\alpha w[t]$, com as mesmas especificações ao anterior; repita o processo para vários valores de β e α e plote os resultados (SNR_m e SNR_i). O que voce observa? Comente.