Universidade Federal da Paraíba
Centro de Energias Alternativas e Renováveis
Departamento de Engenharia Elétrica

Filtros Elétricos

Primeiro Relatório

Prof. Dr. Cleonísio Protásio Souza

Aluno: Vitor de Sousa França - 20180041455

João Pessoa - PB
2024

1 Introdução

Os filtros elétricos são dispositivos utilizados em uma ampla gama de circuitos eletrônicos para diferentes finalidades: em sistemas de comunicação, equipamentos médicos, processamento de sinais, eletrônica de potência, entre outros.

A função dos filtros é permitir ou bloquear diferentes faixas de frequências e dessa forma garantir que o circuito opere como esperado. É utilizando um filtro passa-faixa, por exemplo, que se torna possível selecionar a faixa de frequência desejada em uma emissão de rádio.

Uma utilização muito importante para os filtros é a remoção de ruídos. O ruído pode ser compreendido como um sinal aleatório e indesejado na banda de frequência útil do circuito e é extremamente comum na maioria dos projetos eletrônicos.

Os ruídos podem ser classificados como externos e internos. Os externos são gerados por fontes externas ao equipamento: ruído cósmico, ruído solar, interferência de cabos da rede elétrica próximos ao circuito, são exemplos dessa classe de ruídos. Os ruídos internos, são os gerados por fontes internas ao equipamentos como ruído térmico, cintilação da frequência do cristal entre outros.

Devido a presença quase obrigatória do ruído nos circuitos elétricos, faz-se importante modelá-los e verificar a eficácia dos filtros projetados antes mesmo da concepção final do hardware. Portanto, é indispensável a utilização de ferramentas simulacionais capazes de gerar ruído.

O objetivo desse relatório é citar possíveis *softwares* de simulação capazes de gerar ruído e escolher um para gerar o sinal e o ruído que futuramente serão utilizados para simulação de filtros.

2 *Softwares* de simulação

Os *softwares* de simulação de circuitos são utilizados para planejar e testar diferentes configurações antes da concepção do *hardware*, de tal forma, evitam desperdício de tempo e dinheiro.

Como criar funções aleatórias são muito custosas computacionalmente, simulação de ruídos normalmente são feitas através de uma função pseudo-aleatórias que estão presentes na maioria dos simuladores e linguagens de programação.

Para citar alguns dos *softwares* mais utilizados, o LTSpice permite a criação do ruído através da função ".noise", o matlab através da função "rand" (e variações como "randn" para uma distribuição normal). O Python, linguagem de programação de propósito geral, possui bibliotecas especializadas em cálculo numérico (Numpy) que permite a criação de ruído através do módulo "numpy.random".

Pelo fato de ser uma linguagem de código aberto, gratuita e de propósito geral com bibliotecas bem já bem consolidadas perante a comunidade, o Python foi a linguagem de programação escolhida para criar o ruído e o sinal que estarão presentes nesse relatório e para simular os filtros futuramente.

3 Simulação do sinal e do ruído no Python

O sinal gerado para realização das simulações 1, foi a soma de dois senos com frequências de $f_1 = 20Hz$ e $f_2 = 10Hz$. O sinal de maior frequência possui amplitude $A = 0.5$, enquanto o de menor frequência $A = 1$ A Figura 1 apresenta os sinais no tempo. A frequência de amostragem do sinal é de $f_s = 1000Hz$, o sinal foi capturado durante o tempo de 1 segundo e, portanto, existem mil amostras capturadas $N = 1000$.

$$signal = 0.5\sin(2\pi f_1 t) + \sin(2\pi f_2 t) \tag{1}$$

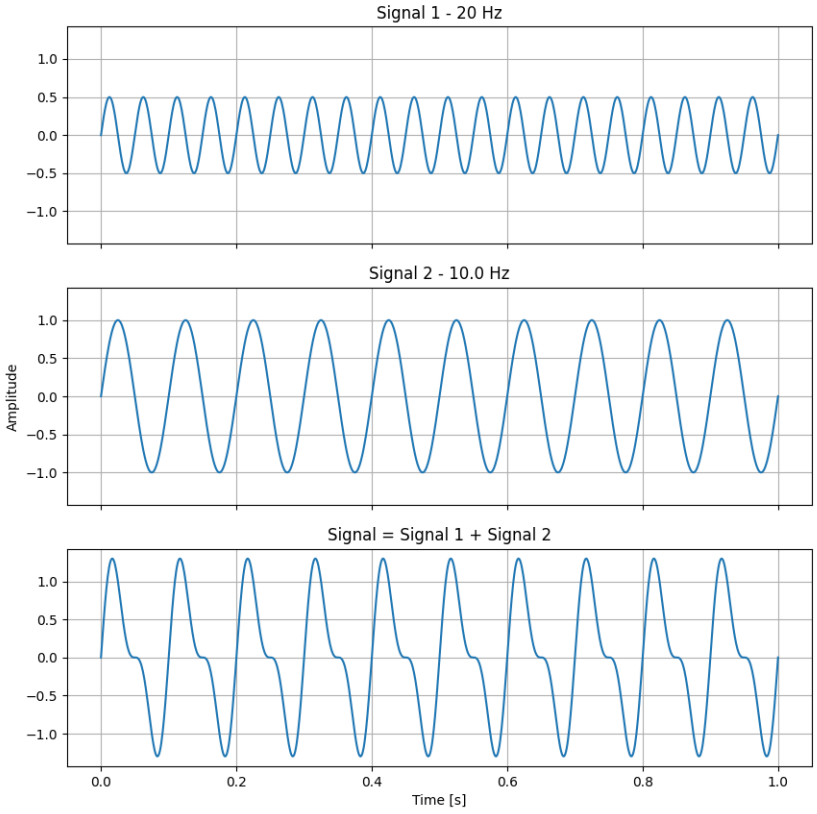

Figura 1: Soma dos senos para gerar o sinal.

O ruído escolhido foi o famoso ruído gaussiano branco. Isto é, um ruído com amplitude definida por uma amostragem da distribuição normal e com presença aproximadamente uniforme em todos os espectros de frequência. Esse ruído foi escolhido por ser uma boa aproximação de vários fenômenos reais, por exemplo, o ruído térmico.

Para gerar esse ruído no Python utilizou-se o método "np.random.normal(mu, sigma, N)", em que a média escolhida foi $mu = 0$, a variância $sigma = 1$, e foram amostrados o mesmo número de pontos que o sinal $N = 1000$.

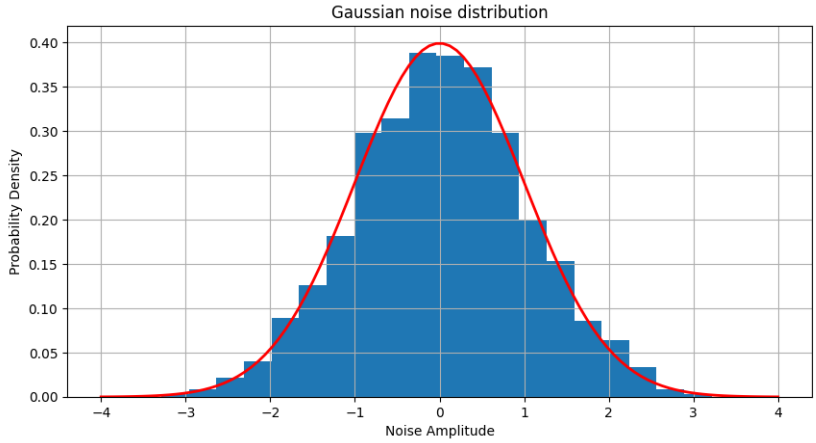

Figura 2: Histograma da distribuição das amostras do ruído, bins=20.

Utilizando o histograma gerado na Figura 2, pode-se verificar a característica da amplitude amostragem que é aproximadamente a distribuição Normal. O sinal do ruído no tempo, bem como a soma do ruído com o sinal podem ser observados através dos gráficos na Figura 3.

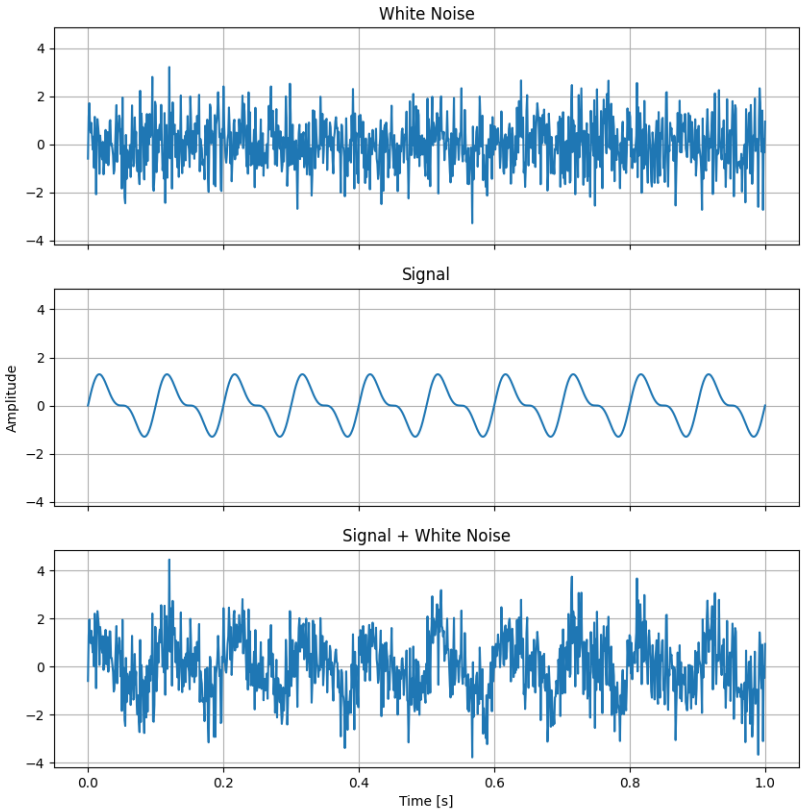

Figura 3: Ruído no tempo e soma do ruído ao sinal.

Para concluir, foi analisada a relação sinal-ruído do sistema em dB, através da fórmula

2. O cálculo da potência foi realizado como a média dos quadrados da amplitude do sinal em cada ponto, resultando em uma potência de sinal de 0,624 W, uma potência de ruído de 1,005 W e uma relação sinal-ruído (SNR) de -2,0689 dB.

$$SNR(dB) = 10 * \log\left(\frac{P_s}{P_n}\right) = -2.0689dB \tag{2}$$

4 Análise dos Sinais no domínio da Frequência

O tratamento de sinais normalmente é realizado no domínio da frequência pois assim é possível decompor um sinal complexo em suas componentes de frequências básicas. Para transformar um sinal no domínio do tempo para o domínio da frequência utiliza-se a transformada de Fourier, no caso de simulações computacionais utiliza-se o algoritmo da transformada rápida de Fourier (sigla em inglês FFT). No Python esse algoritmo pode ser usado através do método "fft" que está no módulo "numpy.fft".

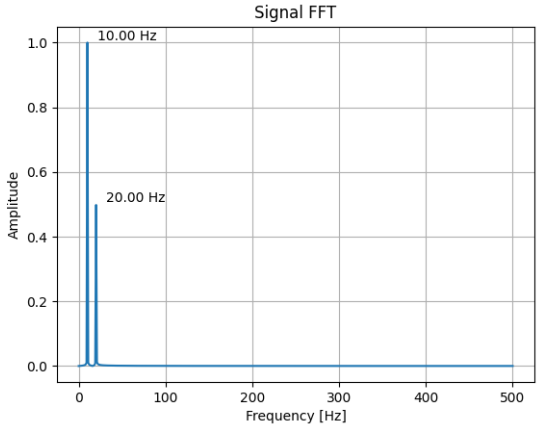

Figura 4: Ruído no tempo e soma do ruído ao sinal.

Foi realizada a avaliação na frequência do sinal sem ruídos, Figura 4 no qual pode-se verificar os dois impulsos nas frequências dos senos, $f_1 = 20Hz$ e $f_2 = 10Hz$.

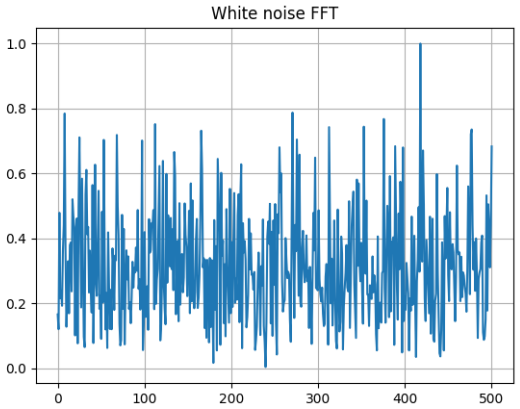

Figura 5: Ruído no tempo e soma do ruído ao sinal.

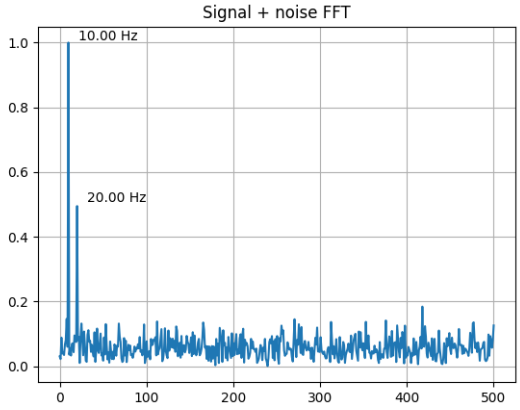

Figura 6: Ruído no tempo e soma do ruído ao sinal.

O espectro de frequências do ruído gaussiano branco, Figura 5, mostra sua presença com amplitudes semelhantes em todas as componentes de frequência. Finalmente o espectro de frequência do sinal com adição do ruído gaussiano branco, Figura 6.

O código referente a esse relatório pode ser encontrado publicamente no repositório do GitHub através do link: https://github.com/V-kr0pt/filtros_ufpb.