

How to normalize a signal for a given SNR

We will normalize a signal such that the Likelihood ratio (LR) test for it has a given signal-to-noise ratio (SNR) in noise with a given Power Spectral Density (PSD). [We often shorten this statement to say: "Normalize the signal to have a given SNR." It is understood in this statement that one is talking about the LR for Gaussian noise as the detection statistic.

Contents

- [Calculation of the norm](#)
- [Test](#)

We will reuse codes that have already been written. Path to folder containing signal and noise generation codes

```
addpath ../DSP/  
addpath ../NOISE/
```

This is the target SNR

```
snr = 10;
```

Data generation parameters

```
nSamples = 2048;  
sampFreq = 1024;  
timeVec = (0:(nSamples-1))/sampFreq;
```

Generate the signal that is to be normalized

```
a1=10;  
a2=3;  
a3=3;  
% Amplitude value does not matter as it will be changed in the normalization  
A = 1;  
sigVec = crcbgenqcsig(timeVec,1,[a1,a2,a3]);
```

We will use the noise PSD used in colGaussNoiseDemo.m but add a constant to remove the parts that are zero. (Exercise: Prove that if the noise PSD is zero at some frequencies but the signal added to the noise is not, then one can create a detection statistic with infinite SNR.)

```
noisePSD = @(f) (f>=100 & f<=300).*(f-100).*(300-f)/10000 + 1;
```

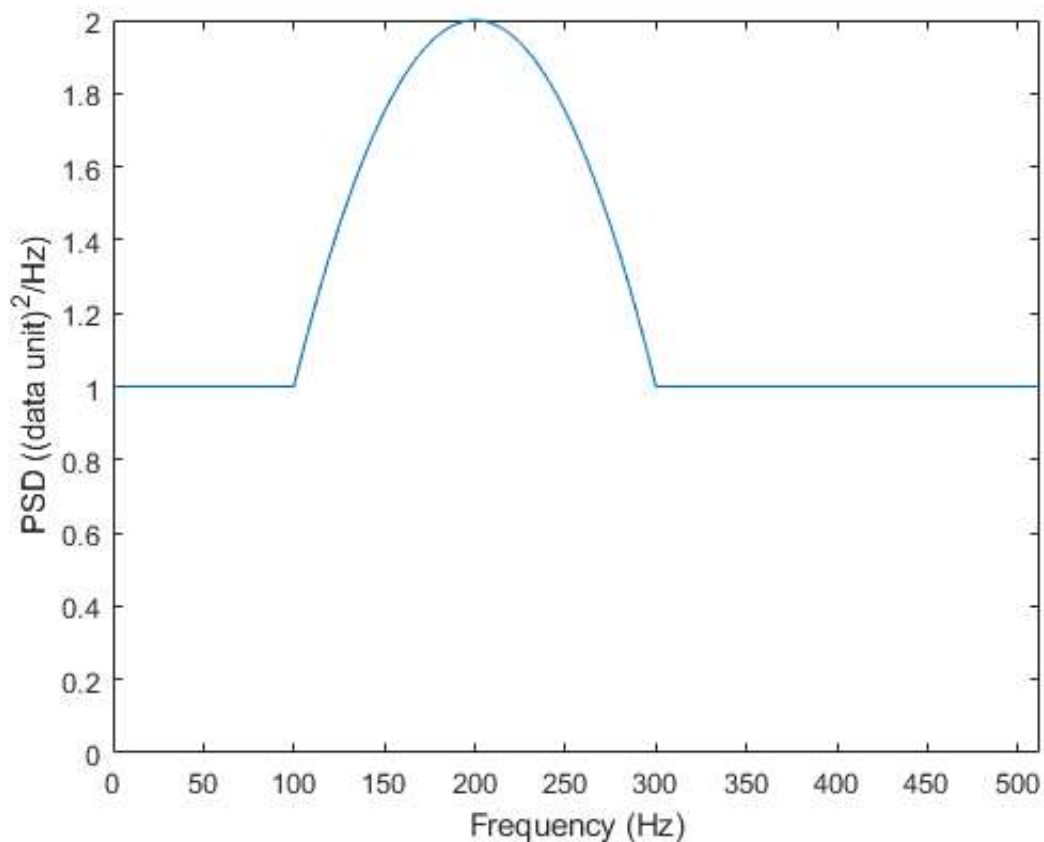
Generate the PSD vector to be used in the normalization. Should be generated for all positive DFT frequencies.

```
dataLen = nSamples/sampFreq;
```

```

kNyq = floor(nSamples/2)+1;
posFreq = (0:(kNyq-1))*(1/dataLen);
psdPosFreq = noisePSD(posFreq);
figure;
plot(posFreq,psdPosFreq);
axis([0,posFreq(end),0,max(psdPosFreq)]);
xlabel('Frequency (Hz)');
ylabel('PSD ((data unit)^2/Hz)');

```



Calculation of the norm

Norm of signal squared is inner product of signal with itself

```

normSigSqr = innerprodpsd(sigVec,sigVec,sampFreq,psdPosFreq);
% Normalize signal to specified SNR
sigVec = snr*sigVec/sqrt(normSigSqr);

```

Test

```

%Obtain LLR values for multiple noise realizations
nH0Data = 1000;
llrH0 = zeros(1,nH0Data);
for lp = 1:nH0Data
    noiseVec = statgaussnoisegen(nSamples,[posFreq(:),psdPosFreq(:)],100,sampFreq);
    llrH0(lp) = innerprodpsd(noiseVec,sigVec,sampFreq,psdPosFreq);
end

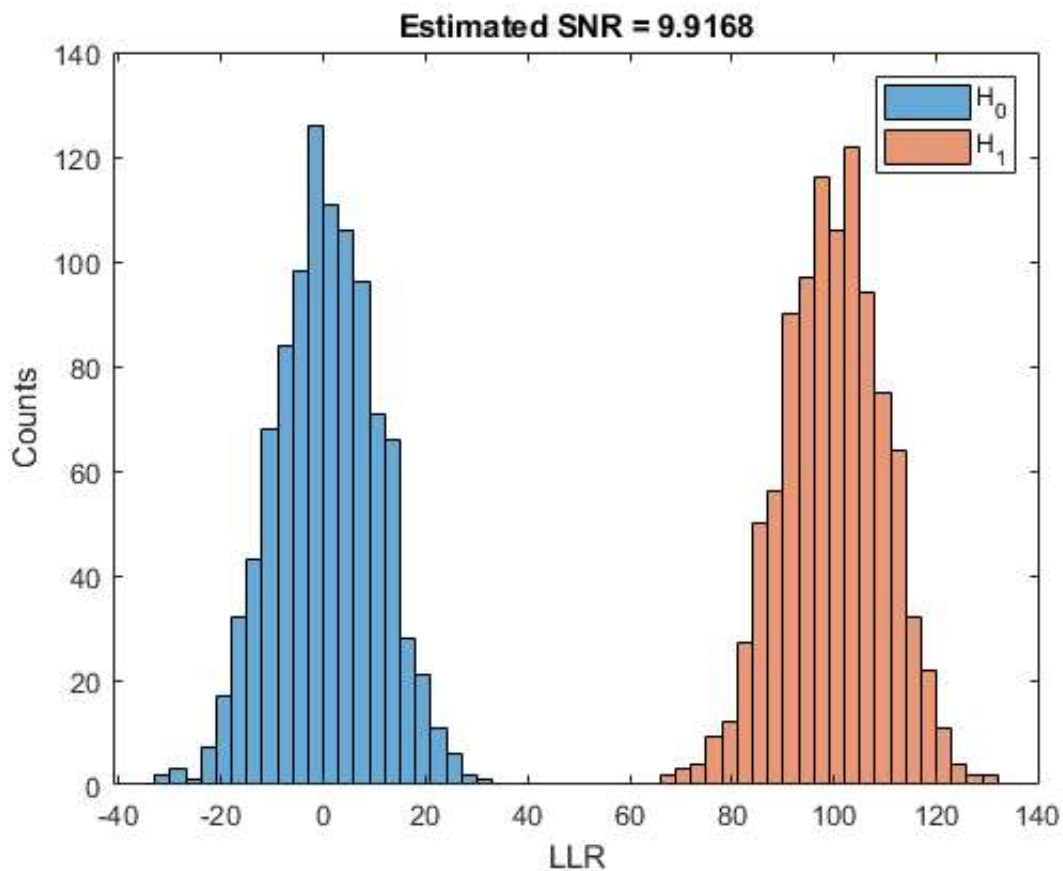
```

```
%Obtain LLR for multiple data (=signal+noise) realizations
nH1Data = 1000;
llrH1 = zeros(1,nH1Data);
for lp = 1:nH0Data
    noiseVec = statgaussnoisegen(nSamples,[posFreq(:),psdPosFreq(:)],100,sampFreq);
    % Add normalized signal
    dataVec = noiseVec + sigVec;
    llrH1(lp) = innerprodpd(dataVec,sigVec,sampFreq,psdPosFreq);
end
```

Signal to noise ratio estimate

```
estSNR = (mean(llrH1)-mean(llrH0))/std(llrH0);

figure;
histogram(llrH0);
hold on;
histogram(llrH1);
xlabel('LLR');
ylabel('Counts');
legend('H_0','H_1');
title(['Estimated SNR = ',num2str(estSNR)]);
```



A data realization

```
figure;
```

```
plot(timeVec,dataVec);  
hold on;  
plot(timeVec,sigVec);  
xlabel('Time (sec)');  
ylabel('Data');
```

