TIMESERIES: THE FINAL CHAPTER

11.9.2018

RECAP

* power spectrum / psd

* spectrogram

* filtering

* nyquist frequency

OVERSAMPLING

* the EEG data from wednesday was originally collected at 22 kHz (22000 samples per second)

* if we had used the original, the data file would have been 1.1 GB instead of 6.5 MB

* and every analysis step that you ran would have taken at least 170x as long

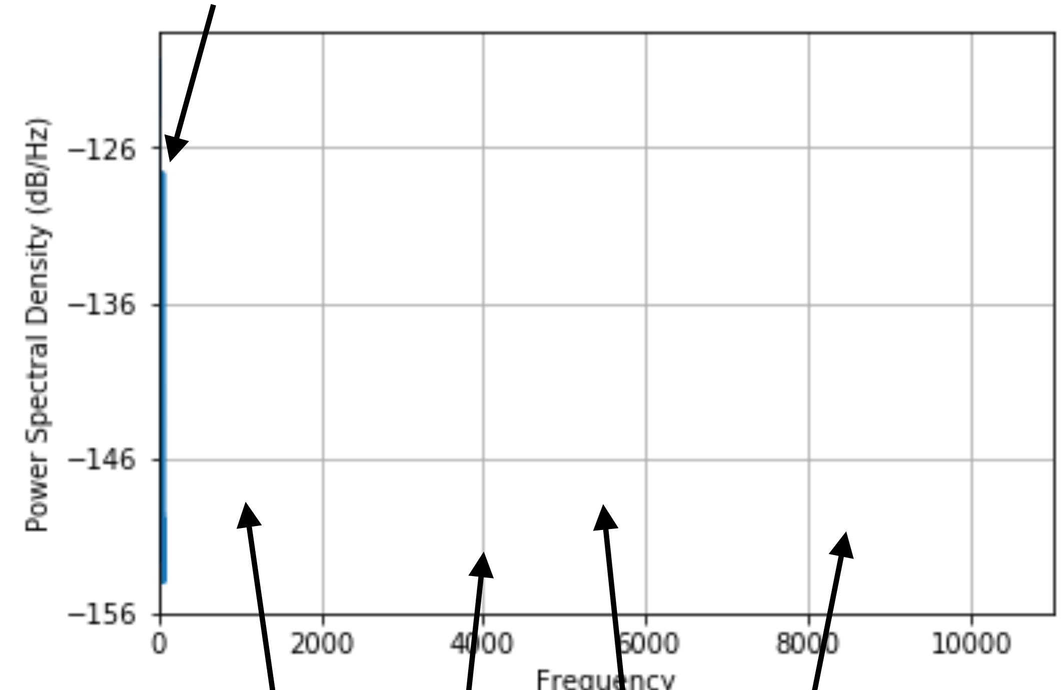
OVERSAMPLING

* would 22 kHz sampling be useful?

* it would increase the Nyquist frequency from 64 Hz to 11000 Hz

* but EEG can't see signals over ~100 Hz

all the interesting stuff



all the extras you get from 22 kHz

OVERSAMPLING

* the point: it’s useful and good to *downsample* EEG signals from 22 kHz to 128 Hz

* so how do we do that?

SUBSAMPLING

* suppose (for simplicity) we have a 20 kHz signal and want to downsample it to 2 kHz

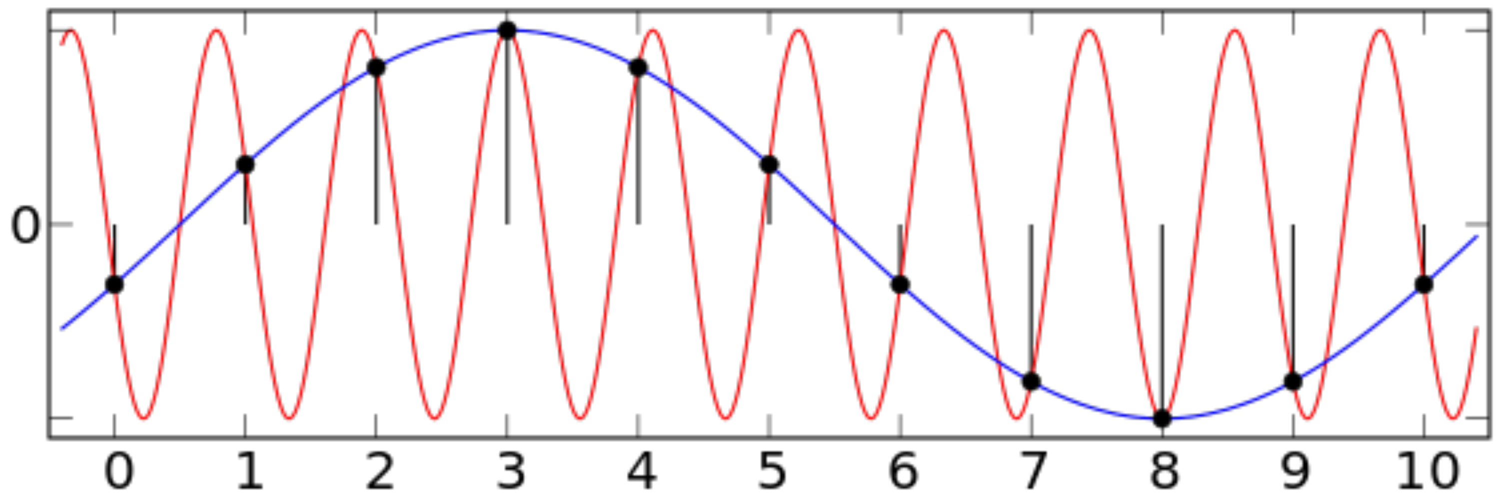
SUBSAMPLING

* one idea: just take every 10th sample!
  * (this is called **subsampling**)
* taking every 10th sample is ***LITERALLY THE WORST IDEA***
  * (let’s see an example)

SUBSAMPLING

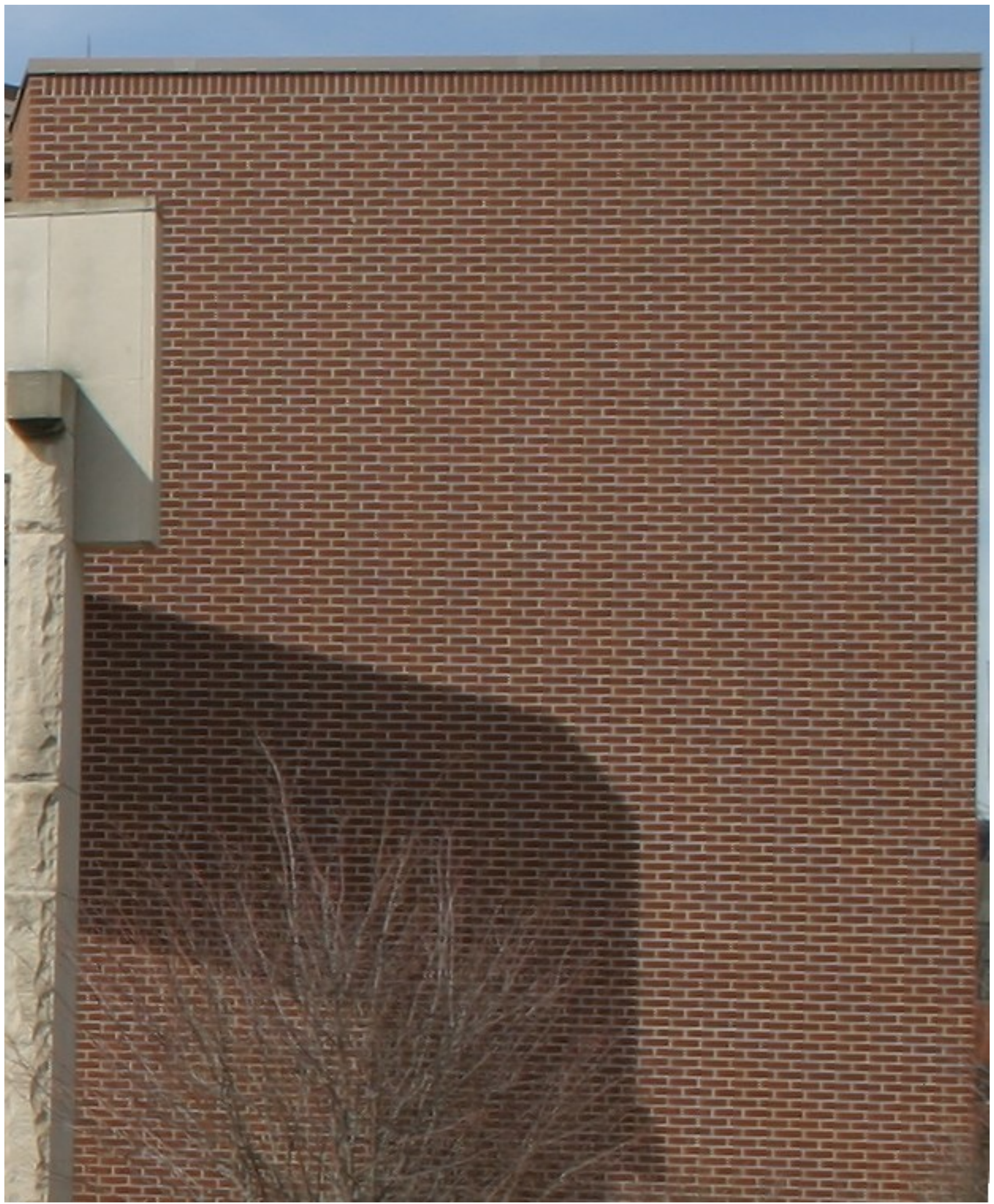
* the weird thing about subsampling is that, instead of removing high frequencies, it *turns them into low frequencies*

* this is called **aliasing**

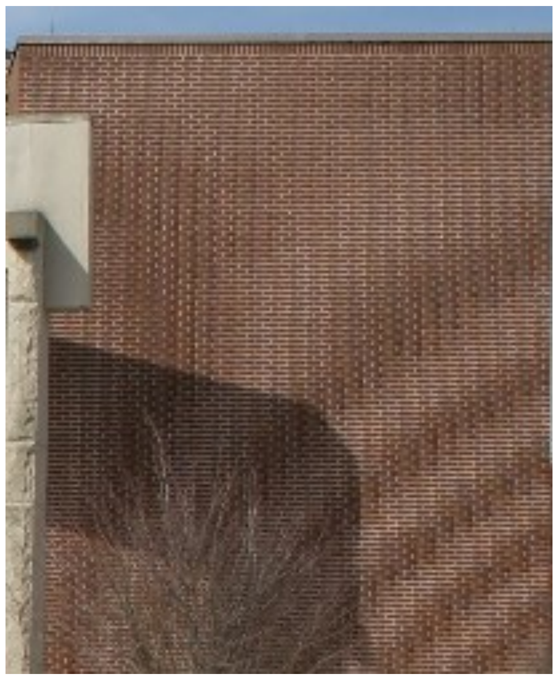


ALIASING IN IMAGES

Original Image



Subsampled

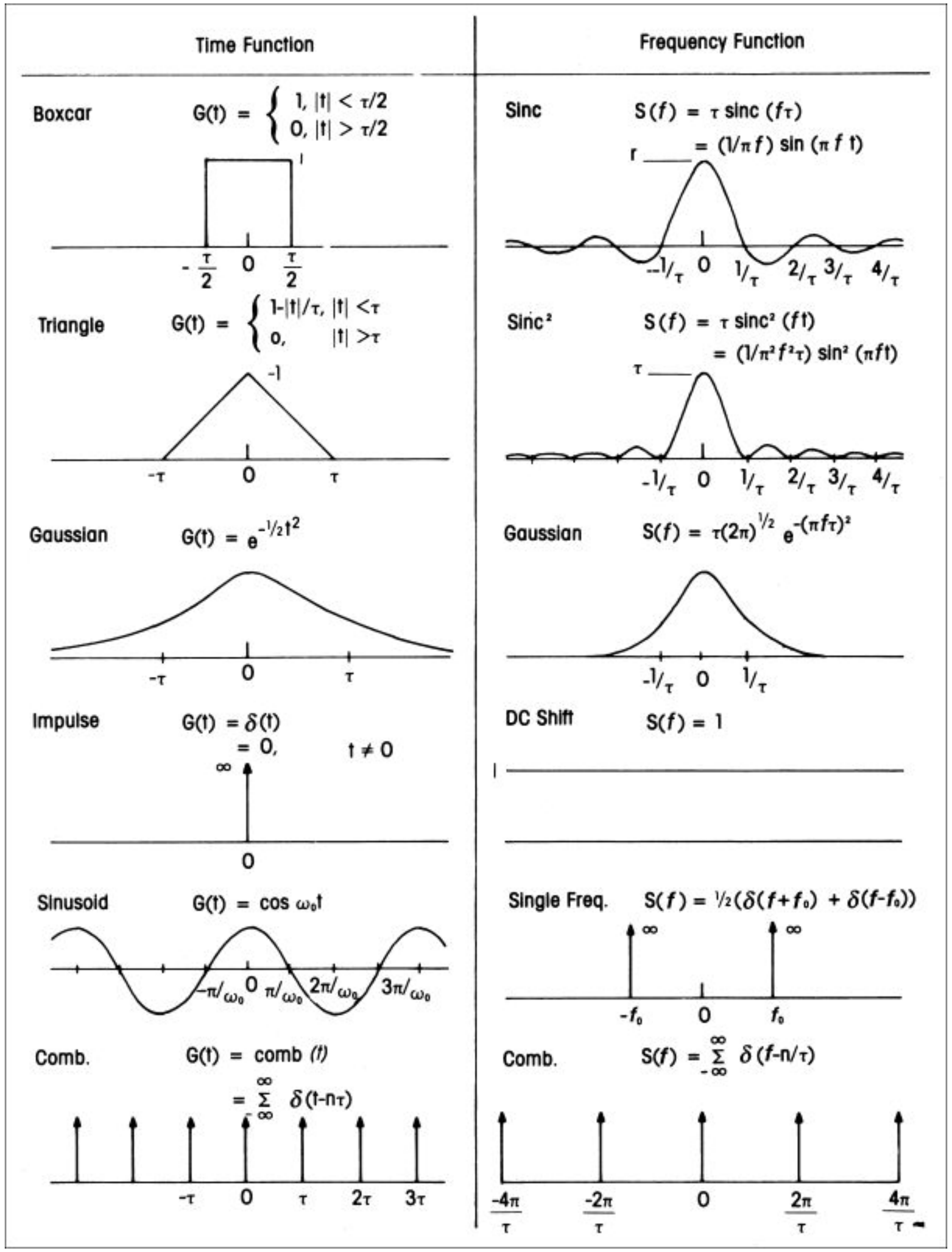


high frequency pattern (bricks) is aliased to low frequency “moiré pattern”

ALIASING

* sampling is like multiplying your timeseries by a "comb" function

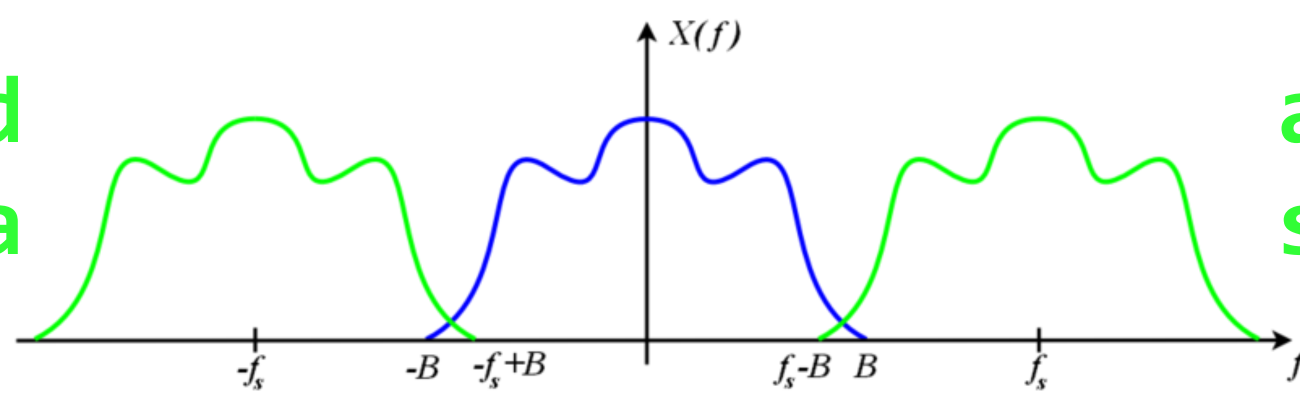
* … which is equivalent to convolving the fourier transform of your timeseries by a comb function



ALIASING

* which means that the fourier transform of the subsampled timeseries can have high frequencies “invading” lower frequencies

aliased spectra



aliased spectra

original spectrum

ANTIALIASING

* how do we solve this?

ANTIALIASING

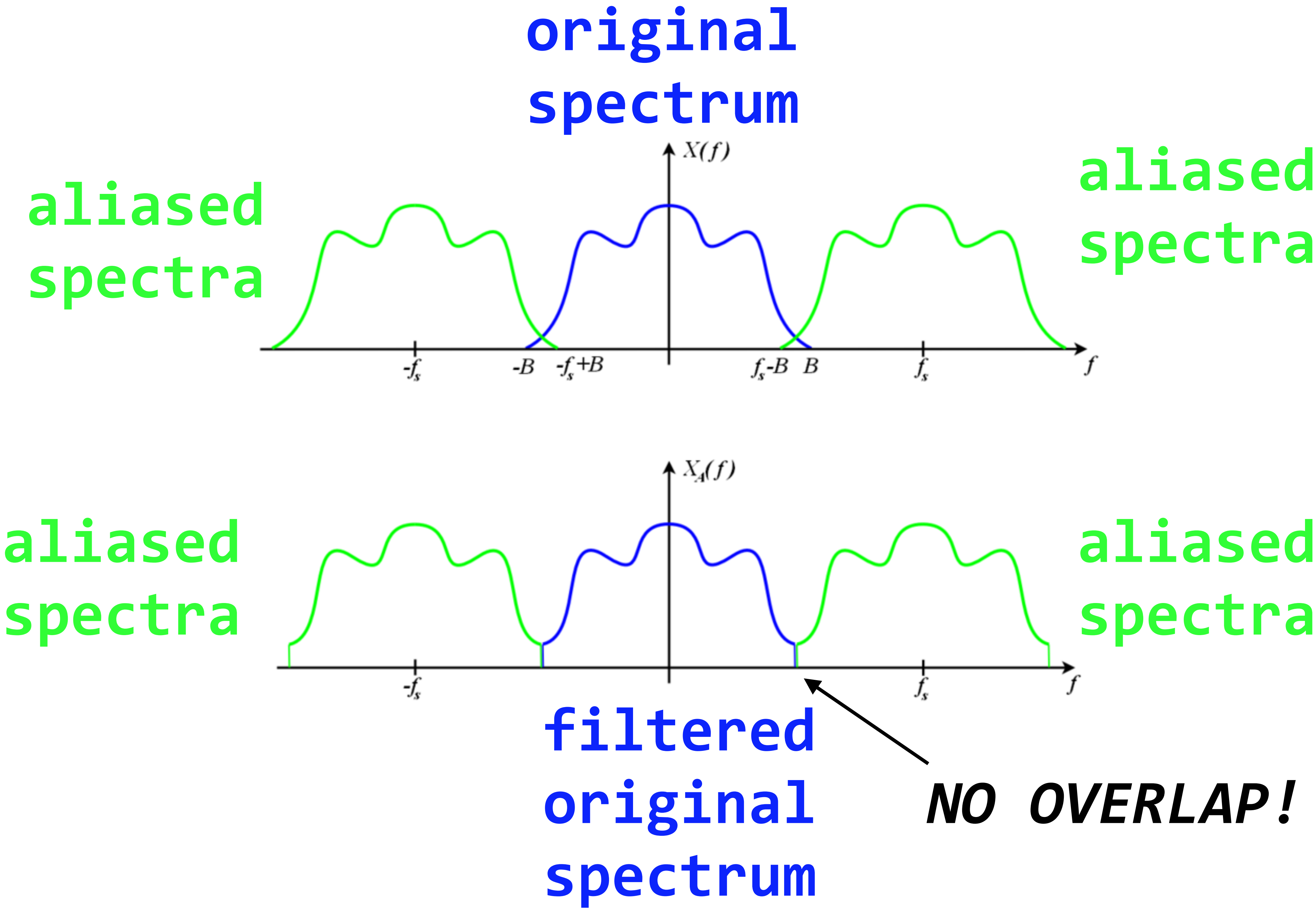
* we can use an **antialiasing** filter

* e.g.: the original signal is sampled at 20 kHz, we want to downsample to 2 kHz

* the new 2 kHz shouldn’t contain any frequencies above Nyquist (1 kHz)

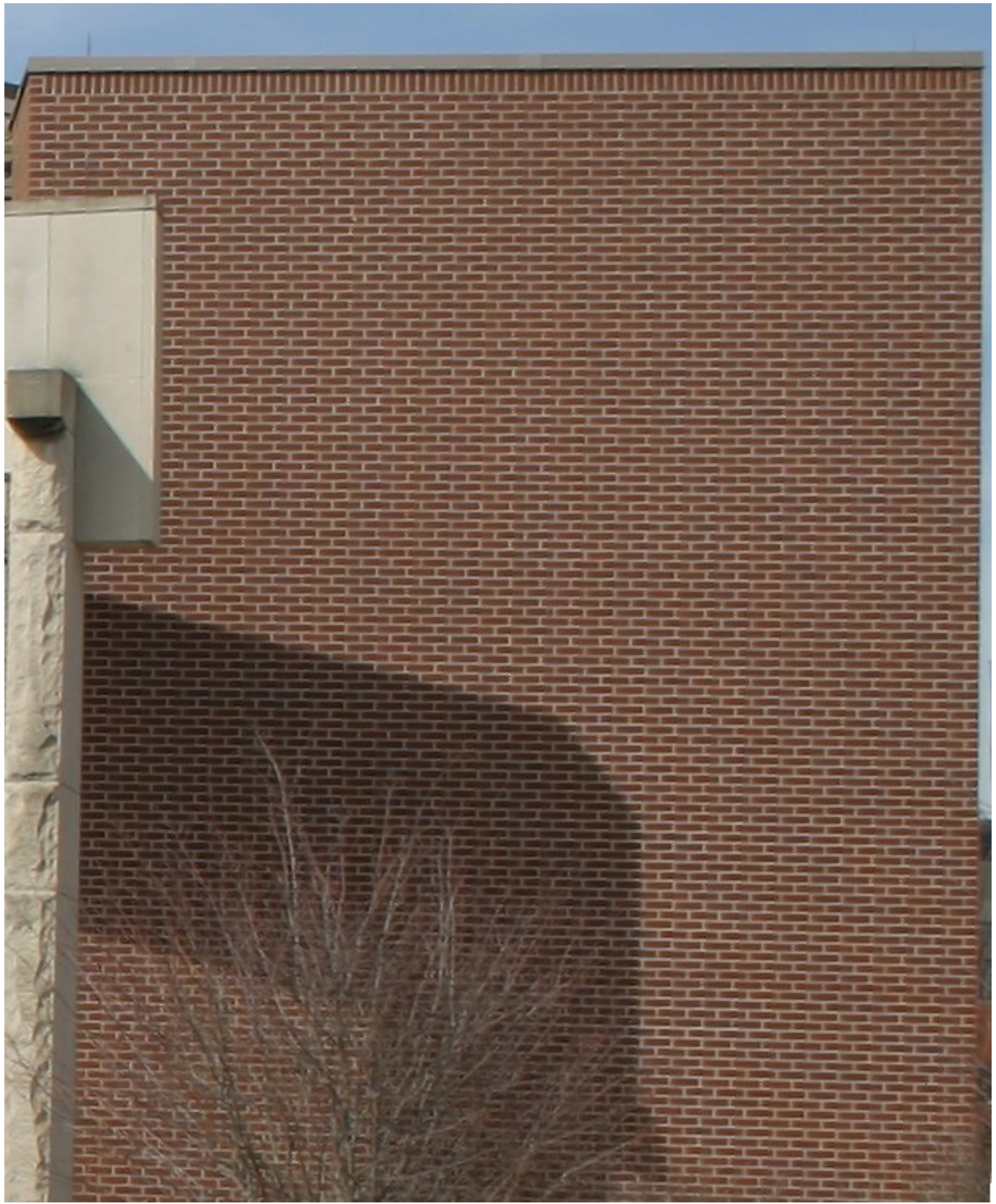
* so we **low-pass filter** the original signal at 1 kHz, and then subsample

ANTIALIASING



ANTIALIASING IN IMAGES

Original Image



Subsampled

Properly downsampled

ANTIALIASING

* there are functions in **scipy.signal** for doing good downsampling/resampling

  * **signal.decimate** is great for downsampling

  * **signal.resample** can do downsampling or upsampling

END