DC removal

To process a WAV file to remove the DC component (frequencies up to 100 Hz) using a time-domain filter and then save the processed signal back to a WAV file, you can use Python with libraries such as scipy and numpy. Here’s a step-by-step guide and example code:

### **Steps**

1. **Read the WAV file**: Use the scipy.io.wavfile module to read the input WAV file.
2. **Design a high-pass filter**: Create a filter that attenuates frequencies below 100 Hz.
3. **Apply the filter to the signal**: Use convolution to apply the high-pass filter to the time-domain signal.
4. **Write the processed signal to a new WAV file**: Save the filtered signal using the scipy.io.wavfile module.

### **Example Code**

[dc\_removal.ipynb](https://colab.research.google.com/drive/1HUMVj3ytaZ8VwZEcg51uxakKoW3gCQoJ?usp=drive_link)

### **Explanation**

1. **Reading the WAV file**: wavfile.read reads the WAV file and returns the sample rate and data.
2. **High-Pass Filter Design**: The butter function from scipy.signal designs a Butterworth high-pass filter. The filtfilt function applies this filter to the data.
3. **DC Removal**: Frequencies below the cutoff frequency (100 Hz) are removed using the high-pass filter.
4. **Normalization**: The filtered data is normalized to avoid clipping when saving to the WAV file.
5. **Writing the WAV file**: The processed data is saved to a new WAV file using wavfile.write.

This script will read an input WAV file, remove the DC component (frequencies below 100 Hz), and save the processed signal to a new WAV file. Make sure to replace input.wav and output.wav with the actual paths to your files.

### **Explanation**

The only part that needs a mathematical explanation here is [high\_pass\_filter](https://docs.google.com/document/d/1TMX2BVJsb9pR2u-Hjje4Dzmeyi1PreCUvln-jKwZDtY/edit?usp=drive_link)