hw5

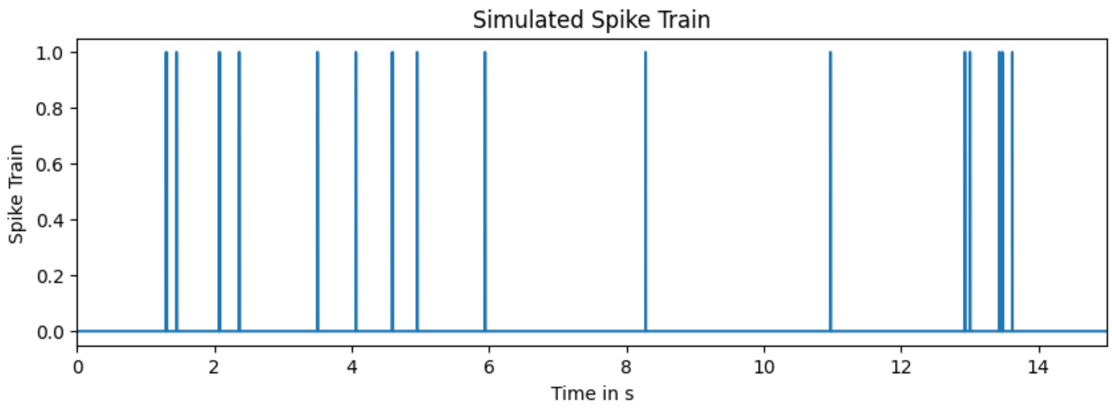
February 28, 2024

1 Deconvultion

1.1 Part a

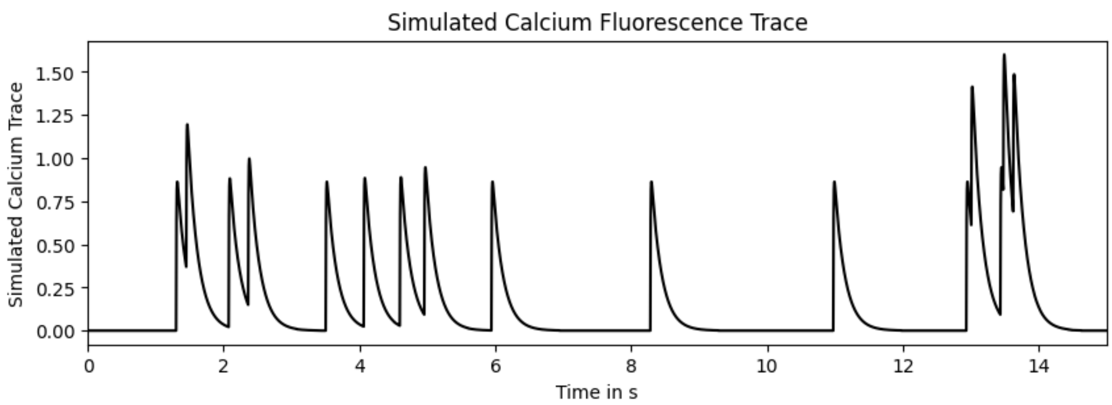
[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from scipy.fft import fft, fftfreq, ifft
```

[2]:
```
T = 15
Fs = 2000
p = 0.0003
N = T * Fs

np.random.seed(42)
spike_train = (np.random.rand(N) < p).astype(int)
time_s = np.arange(N) / Fs

plt.figure(figsize = (10, 3))
plt.plot(time_s, spike_train, drawstyle = 'steps-pre')
plt.xlabel('Time in s')
plt.ylabel('Spike Train')
plt.title('Simulated Spike Train')
plt.xlim(0, T)
plt.show()
```



1.2 Part b

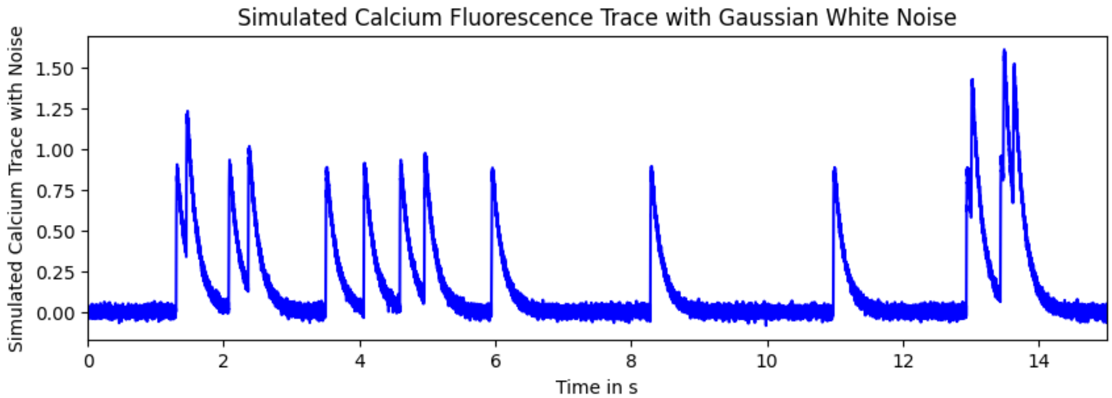
[3]:
```
def impulse_response(t, tau1 = 0.005, tau2 = 0.15):
    double_el_ir_function = (t >= 0) * (1 - np.exp(-t / tau1)) * np.exp(-t /␣
 ↪tau2)
    return double_el_ir_function

delta_t = 1/Fs
time_impulse_response = np.arange(-1, 1, delta_t)
h = impulse_response(time_impulse_response)
simulated_calcium_trace = np.convolve(spike_train, h, mode = 'same')

plt.figure(figsize = (10, 3))
plt.plot(time_s, simulated_calcium_trace, color = 'black')
plt.xlabel('Time in s')
plt.ylabel('Simulated Calcium Trace')
plt.title('Simulated Calcium Fluorescence Trace')
plt.xlim(0, T)
plt.show()
```



1.3 Part c

[4]:
```
noise_rms = 0.02
noise = noise_rms * np.random.randn(N)
simulated_calcium_trace_noise = simulated_calcium_trace + noise

plt.figure(figsize = (10, 3))
plt.plot(time_s, simulated_calcium_trace_noise, color = 'blue')
plt.xlabel('Time in s')
plt.ylabel('Simulated Calcium Trace with Noise')
```

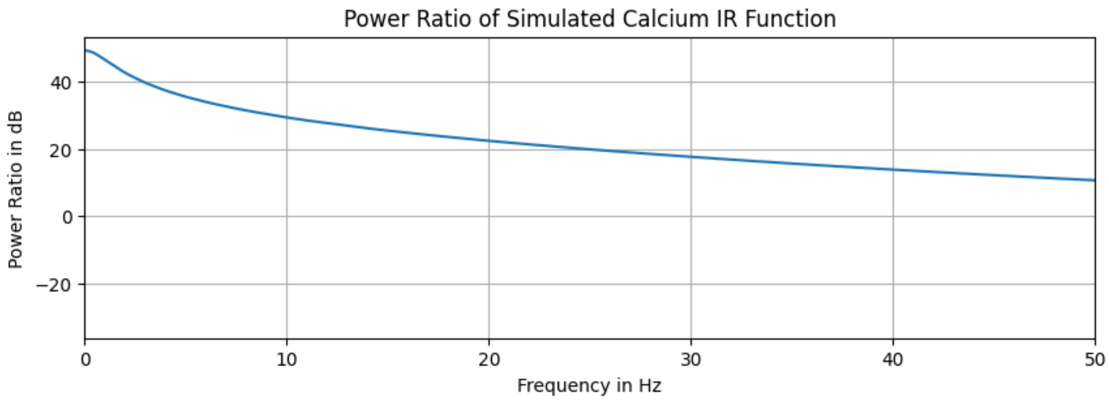
```
plt.title('Simulated Calcium Fluorescence Trace with Gaussian White Noise')
plt.xlim(0, T)
plt.show()
```



1.4 Part d

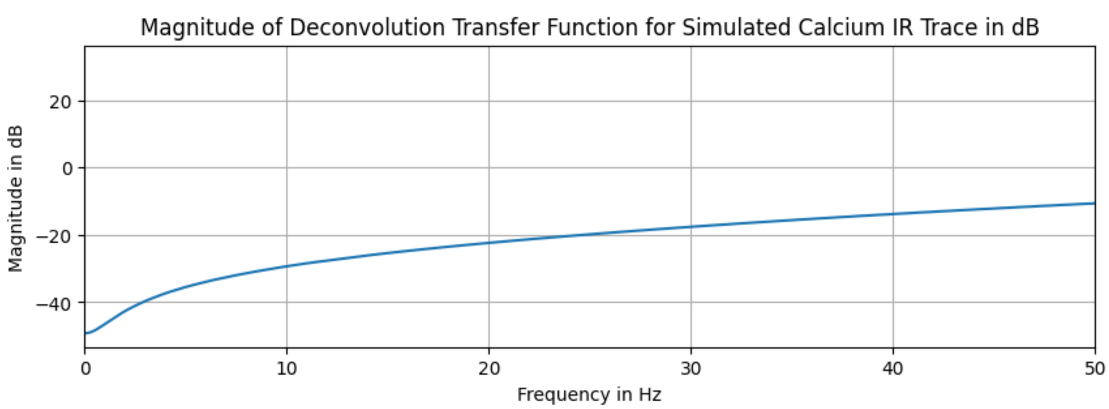
low-pass filter

[5]:
```
ft_ht = fft(h, n = N)
frequencies = fftfreq(N, d = delta_t)
power_ratio = np.abs(ft_ht)**2
dB_of_power_ratio = 10 * np.log10(power_ratio)

plt.figure(figsize = (10, 3))
plt.plot(frequencies[:N//2], dB_of_power_ratio[:N//2])
plt.xlim(0, 50)
plt.xlabel('Frequency in Hz')
plt.ylabel('Power Ratio in dB')
plt.title('Power Ratio of Simulated Calcium IR Function')
plt.grid(True)
plt.show()
```



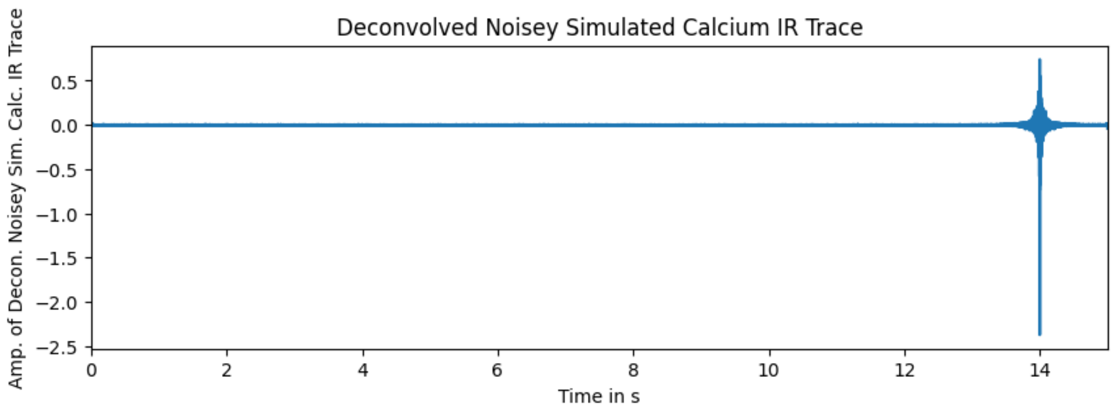
1.5 Part e

[6]:
```
D_w = 1 / ft_ht
D_w_squared = np.abs(D_w)**2
D_w_squared_dB = 10 * np.log10(D_w_squared)

plt.figure(figsize = (10, 3))
plt.plot(frequencies[:N//2], D_w_squared_dB[:N//2])
plt.xlim(0, 50)
plt.xlabel('Frequency in Hz')
plt.ylabel('Magnitude in dB')
plt.title('Magnitude of Deconvolution Transfer Function for Simulated Calcium␣
↪IR Trace in dB')
plt.grid(True)
plt.show()
```



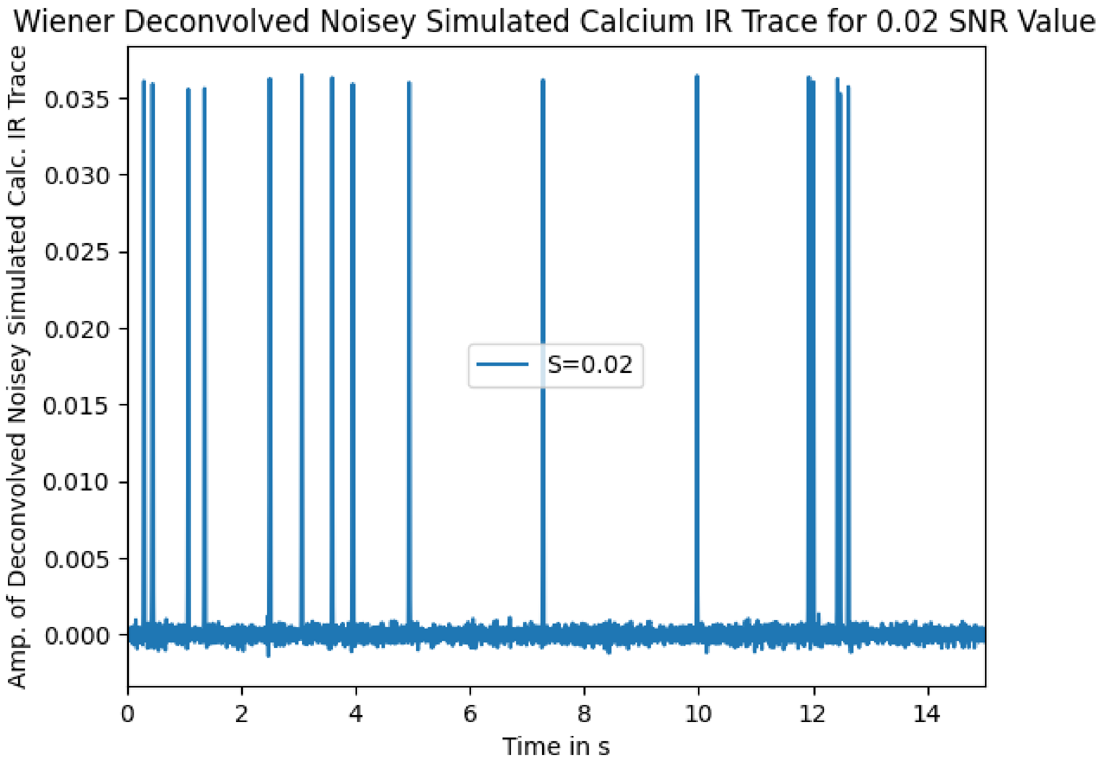
1.6 Part f

[7]:
```
sc_deconvolved_signal = ifft(simulated_calcium_trace_noise * D_w).real

plt.figure(figsize = (10, 3))
plt.plot(time_s, sc_deconvolved_signal)
plt.xlim(0, T)
plt.xlabel('Time in s')
plt.ylabel('Amp. of Decon. Noisey Sim. Calc. IR Trace')
plt.title('Deconvolved Noisey Simulated Calcium IR Trace')
plt.show()
```

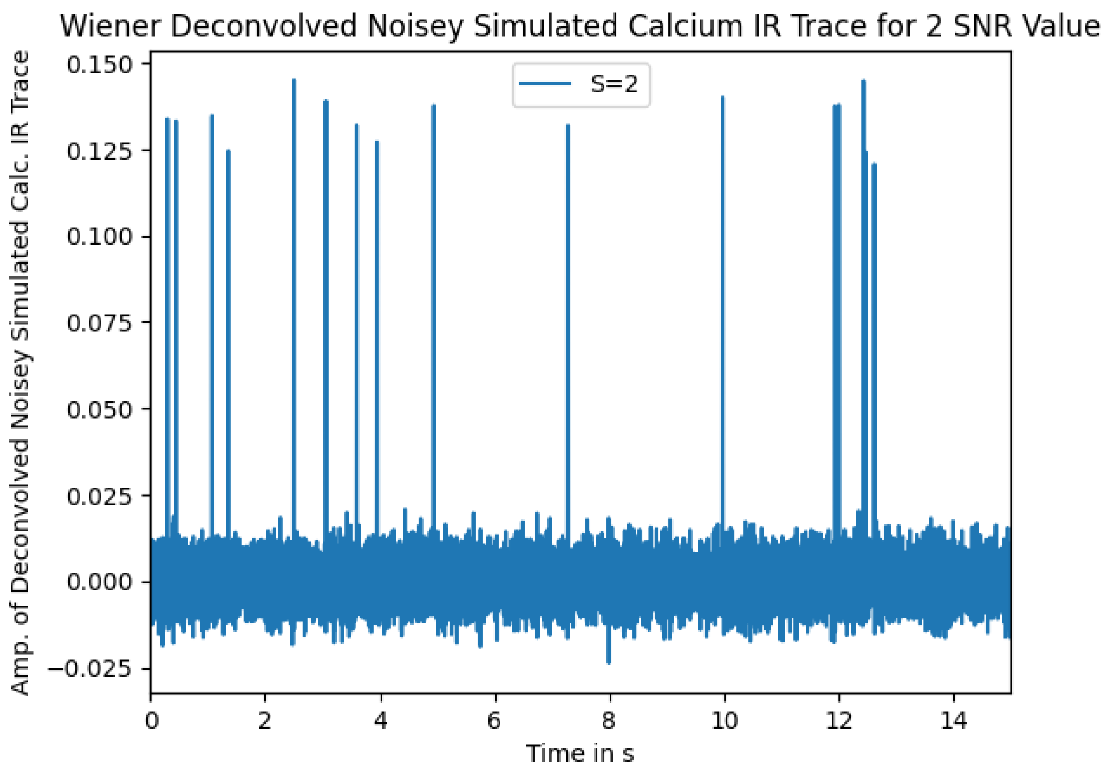


1.7 Part g

[8]:
```
snr = [0.02]

for snr_index in snr:

    D_wiener = np.conj(ft_ht) / (np.abs(ft_ht)**2 + 1/snr_index)
    sc_deconvolved_signal_wiener_fft = fft(simulated_calcium_trace_noise, n =␣
↪N) * D_wiener
    sc_deconvolved_signal_wiener = ifft(sc_deconvolved_signal_wiener_fft).real
    plt.plot(time_s, sc_deconvolved_signal_wiener, label = f"S={snr_index}")

plt.xlabel('Time in s')
plt.ylabel('Amp. of Deconvolved Noisey Simulated Calc. IR Trace')
plt.title('Wiener Deconvolved Noisey Simulated Calcium IR Trace for 0.02 SNR␣
↪Value')
plt.legend()
plt.xlim(0, T)
plt.show()
```

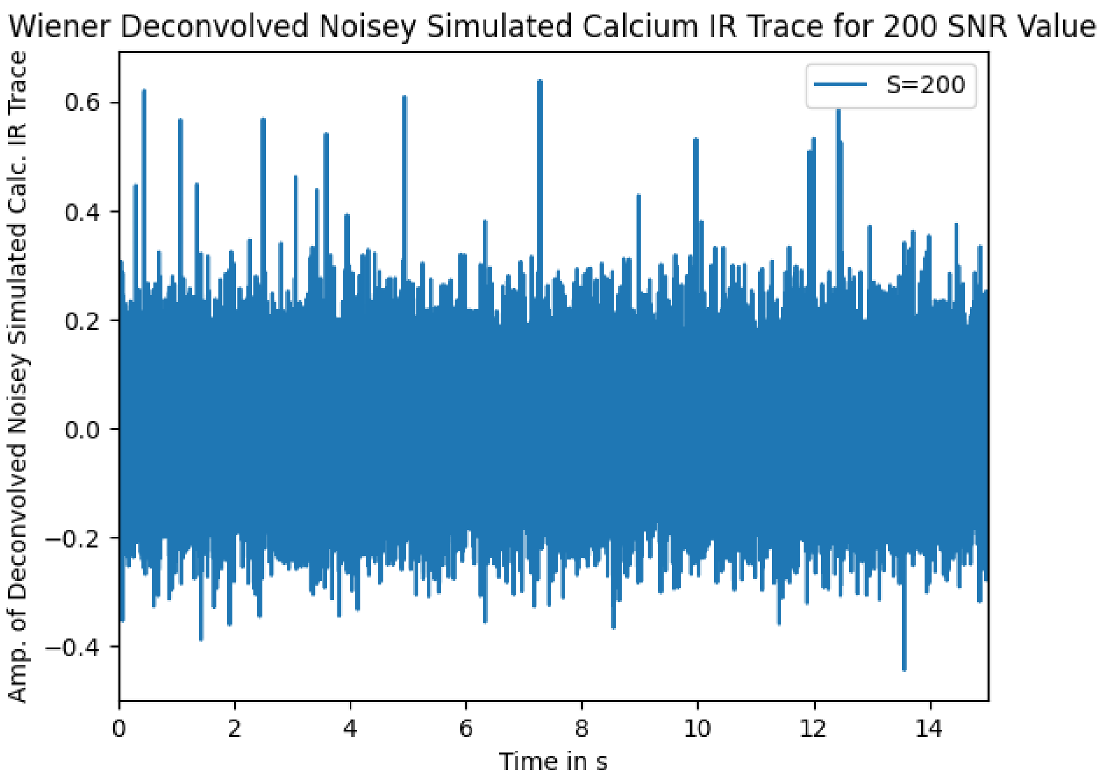


[9]:
```
snr = [2]

for snr_index in snr:

    D_wiener = np.conj(ft_ht) / (np.abs(ft_ht)**2 + 1/snr_index)
    sc_deconvolved_signal_wiener_fft = fft(simulated_calcium_trace_noise, n =␣
 ↪N) * D_wiener
    sc_deconvolved_signal_wiener = ifft(sc_deconvolved_signal_wiener_fft).real
    plt.plot(time_s, sc_deconvolved_signal_wiener, label = f"S={snr_index}")

plt.xlabel('Time in s')
plt.ylabel('Amp. of Deconvolved Noisey Simulated Calc. IR Trace')
plt.title('Wiener Deconvolved Noisey Simulated Calcium IR Trace for 2 SNR␣
 ↪Value')
plt.legend()
plt.xlim(0, T)
plt.show()
```

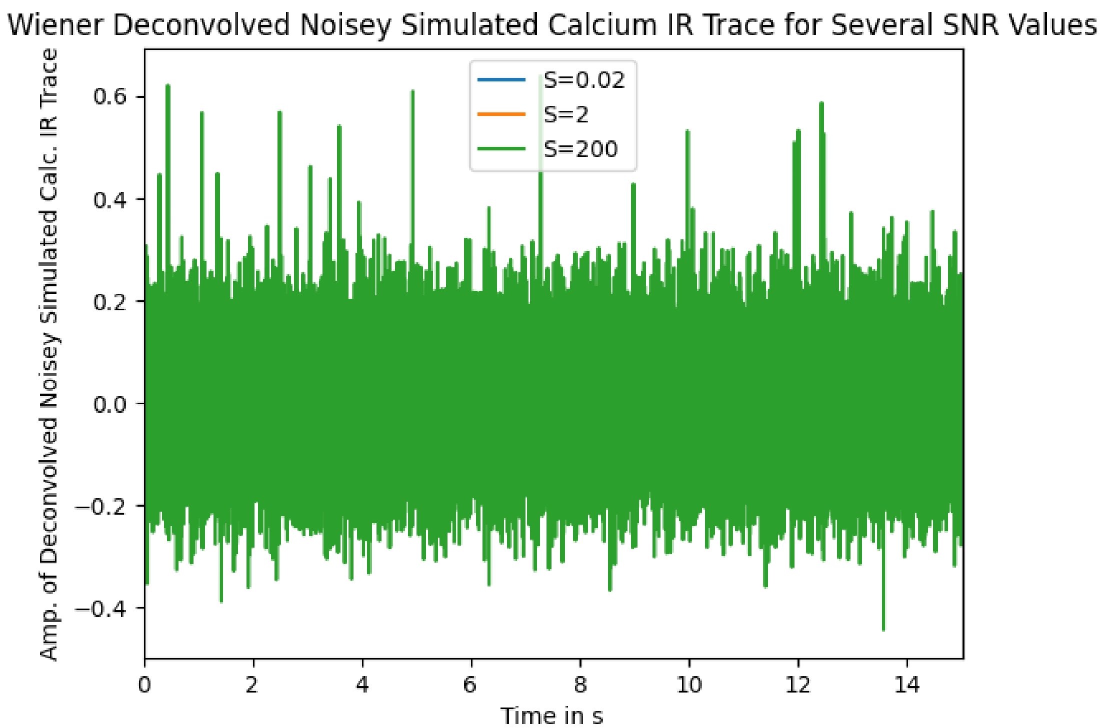


[10]:

```
snr = [200]

for snr_index in snr:

    D_wiener = np.conj(ft_ht) / (np.abs(ft_ht)**2 + 1/snr_index)
    sc_deconvolved_signal_wiener_fft = fft(simulated_calcium_trace_noise, n =␣
 ↪N) * D_wiener
    sc_deconvolved_signal_wiener = ifft(sc_deconvolved_signal_wiener_fft).real
    plt.plot(time_s, sc_deconvolved_signal_wiener, label=f"S={snr_index}")

plt.xlabel('Time in s')
plt.ylabel('Amp. of Deconvolved Noisey Simulated Calc. IR Trace')
plt.title('Wiener Deconvolved Noisey Simulated Calcium IR Trace for 200 SNR␣
 ↪Value')
plt.legend()
plt.xlim(0, T)
plt.show()
```



[11]:
```
snr = [0.02, 2, 200]

for snr_index in snr:
    D_wiener = np.conj(ft_ht) / (np.abs(ft_ht)**2 + 1/snr_index)
    sc_deconvolved_signal_wiener_fft = fft(simulated_calcium_trace_noise, n =␣
↪N) * D_wiener
    sc_deconvolved_signal_wiener = ifft(sc_deconvolved_signal_wiener_fft).real
    plt.plot(time_s, sc_deconvolved_signal_wiener, label=f"S={snr_index}")

plt.xlabel('Time in s')
plt.ylabel('Amp. of Deconvolved Noisey Simulated Calc. IR Trace')
plt.title('Wiener Deconvolved Noisey Simulated Calcium IR Trace for Several SNR␣
↪Values')
plt.legend()
plt.xlim(0, T)
plt.show()
```


Wiener Deconvolved Noisey Simulated Calcium IR Trace for Several SNR Values
Amp. of Deconvolved Noisey Simulated Calc. IR Trace
Time in s
S=0.02
S=2
S=200
0.6
0.4
0.2
0.0
−0.2
−0.4
0
2
4
6
8
10
12
14