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

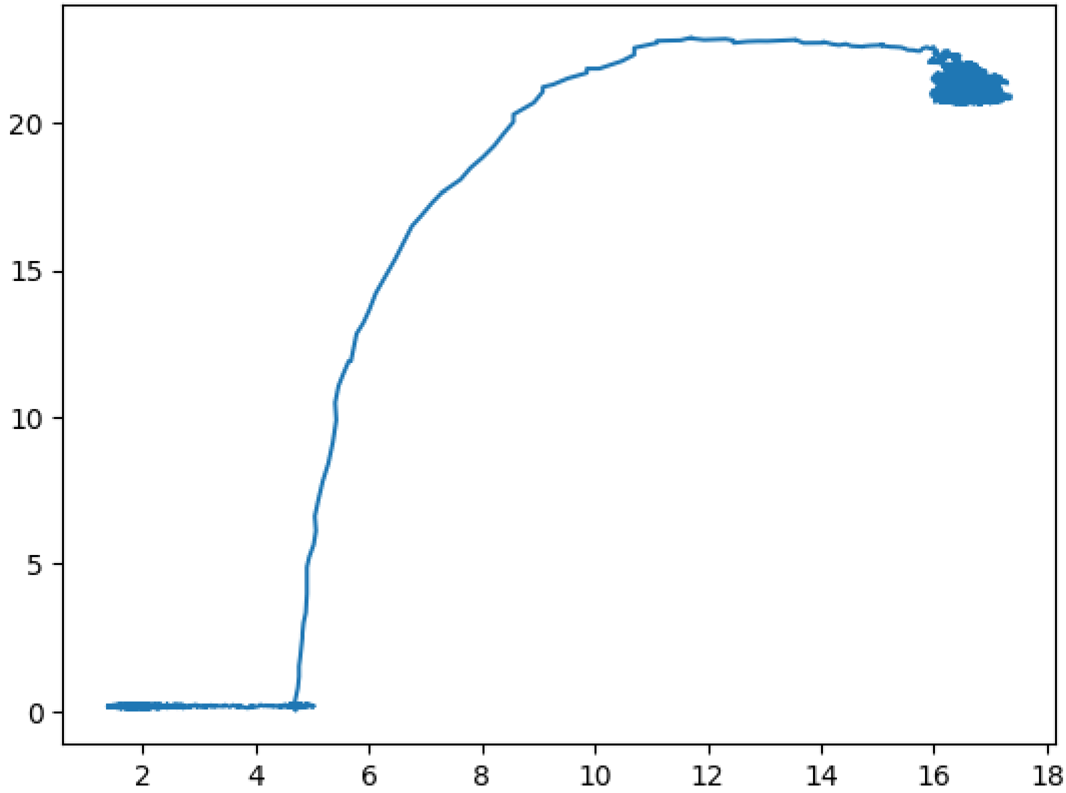
```
In [2]: ls

        23_16:8_data.csv  Untitled.ipynb
```

```
In [47]: hallRaw_df = pd.read_csv('23_16:8_data.csv', header = None)
```

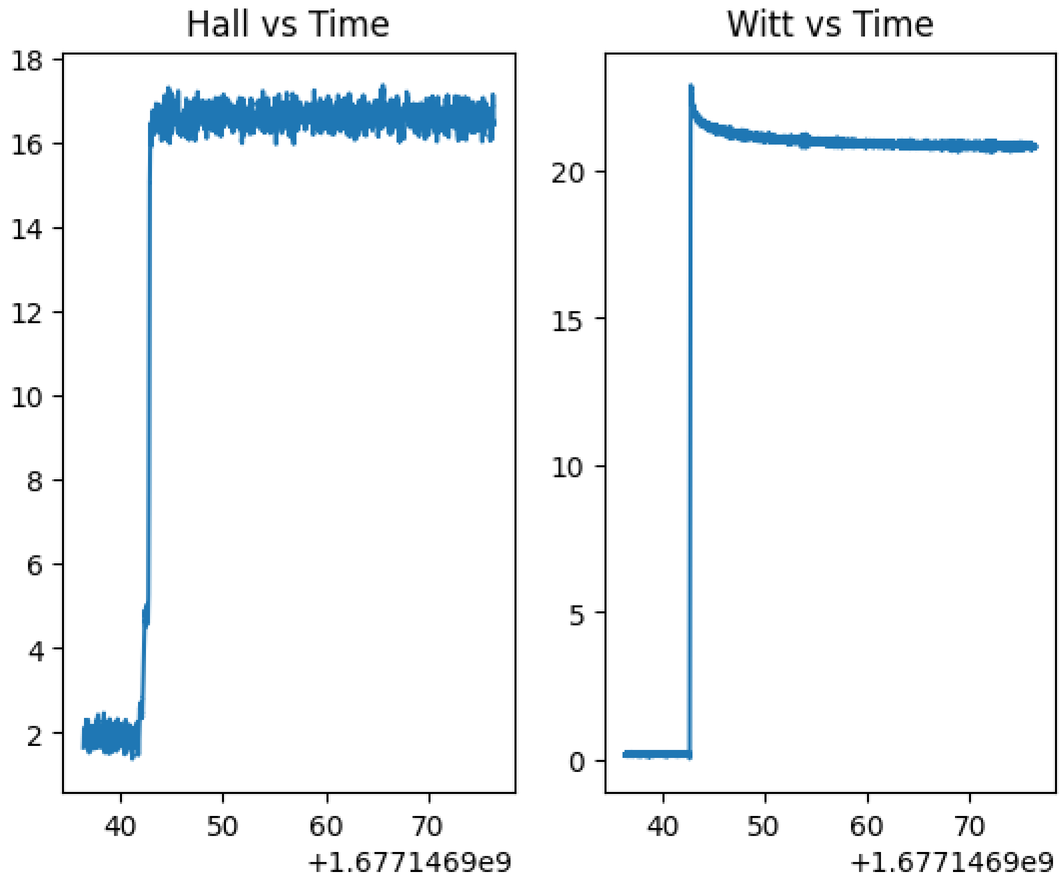
```
In [48]: hallRaw_df.rename(columns={0: 'time', 1: 'witt', 2:'hall'}, inplace=True)
```

```
In [49]: hall = abs(hallRaw_df['hall'])
         witt= abs(hallRaw_df['witt'])
         time = hallRaw_df['time']
```

```
In [6]: plot(hall,witt)

Out[6]: [<matplotlib.lines.Line2D at 0x7fbf83ee74c0>]
```

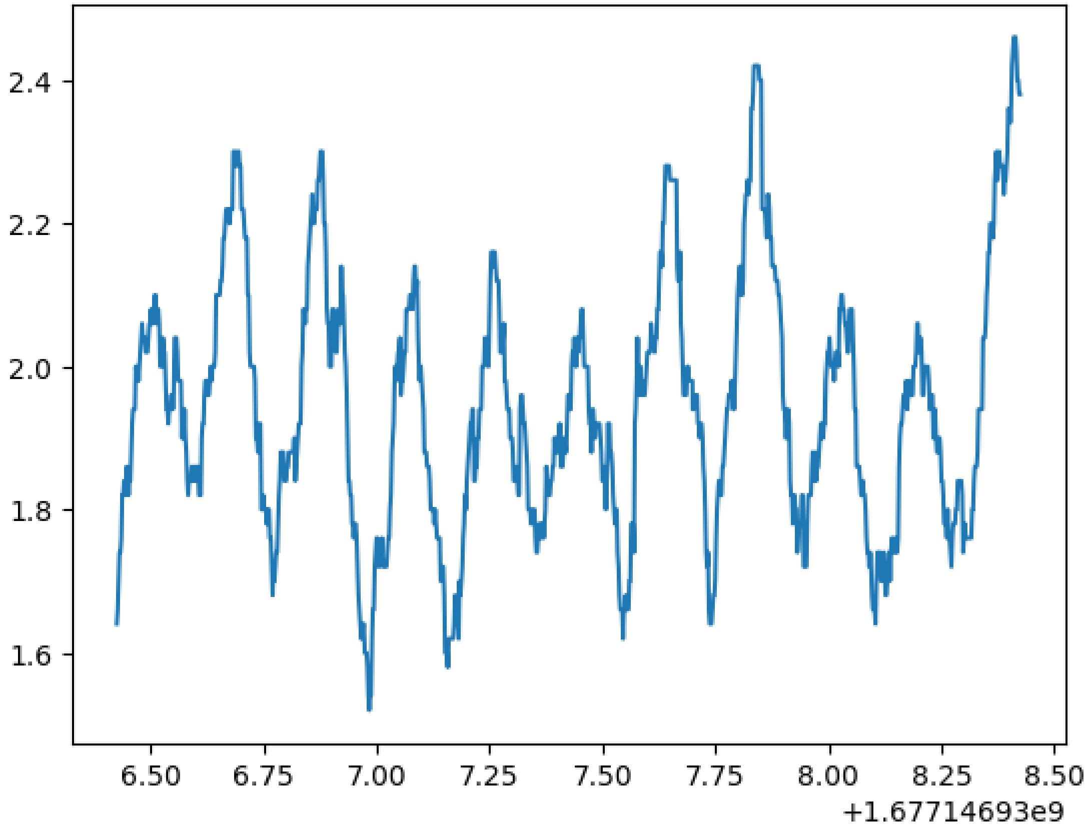


```
In [7]: figure, axis = plt.subplots(1,2)

        axis[0].plot(time,hall)
        axis[0].set_title("Hall vs Time")

        # For Cosine Function
        axis[1].plot(time, witt)
        axis[1].set_title("Witt vs Time")

Out[7]: Text(0.5, 1.0, 'Witt vs Time')
```



In [8]:
```
plt.plot(time[:1000],hall[:1000])
```

Out[8]:
```
[<matplotlib.lines.Line2D at 0x7fbfe0d9e080>]
```



Bandpass filter

In [50]:
```
from scipy.signal import butter,filtfilt
T = 20          # Sample Period
fs = 495.0        # sample rate, Hz
cutoff = 3       # desired cutoff frequency of the filter, Hz ,      slightly higher t
nyq = 0.5 * fs   # Nyquist Frequency
```

```
order = 2       # sin wave can be approx represented as quadratic
n = int(T * fs) # total number of samples
```

In [51]:

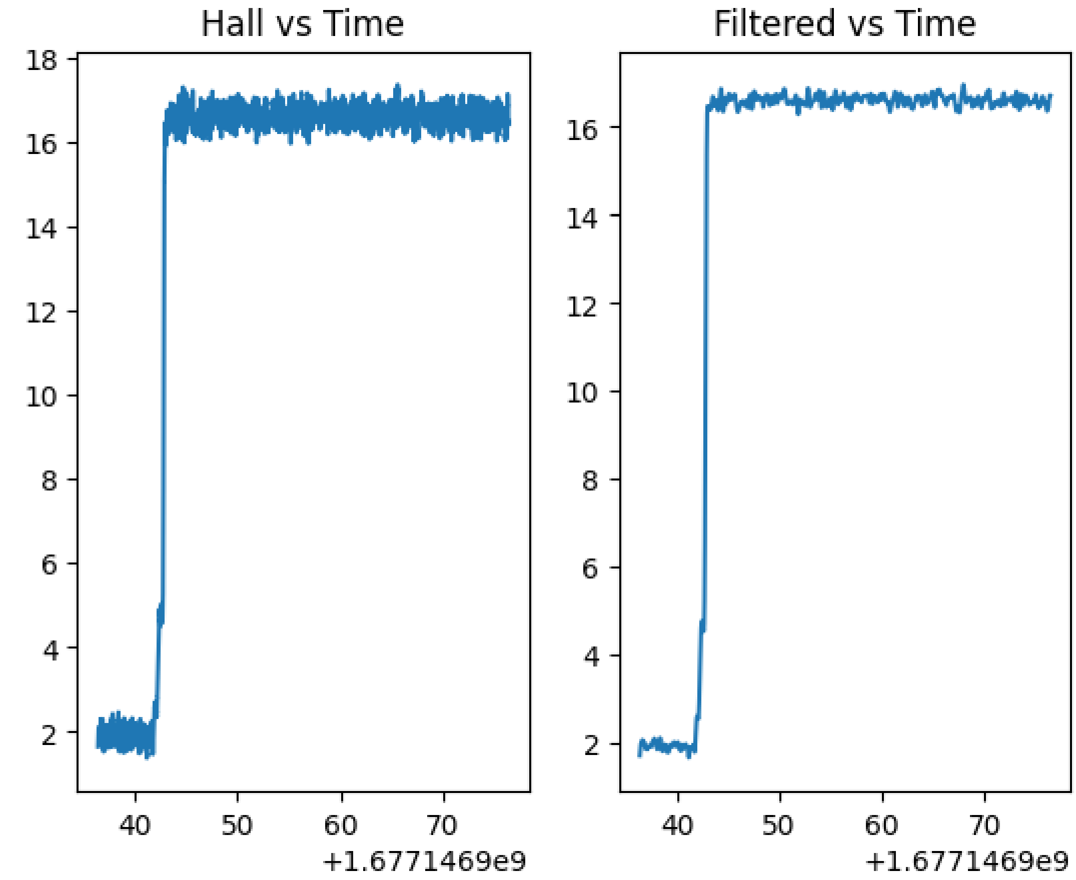
```
def butter_lowpass_filter(data, cutoff, fs, order):
    normal_cutoff = cutoff / nyq
    # Get the filter coefficients
    b, a = butter(order, normal_cutoff, btype='low', analog=False)
    y = filtfilt(b, a, data)
    return y
```

In [52]:

```
hall_filtered = butter_lowpass_filter(hall, cutoff, fs, order)
figure, axis = plt.subplots(1,2)

axis[0].plot(time,hall)
axis[0].set_title("Hall vs Time")

# For Cosine Function
axis[1].plot(time, hall_filtered)
axis[1].set_title("Filtered vs Time")
```

Out[52]:

```
Text(0.5, 1.0, 'Filtered vs Time')
```



In [32]:

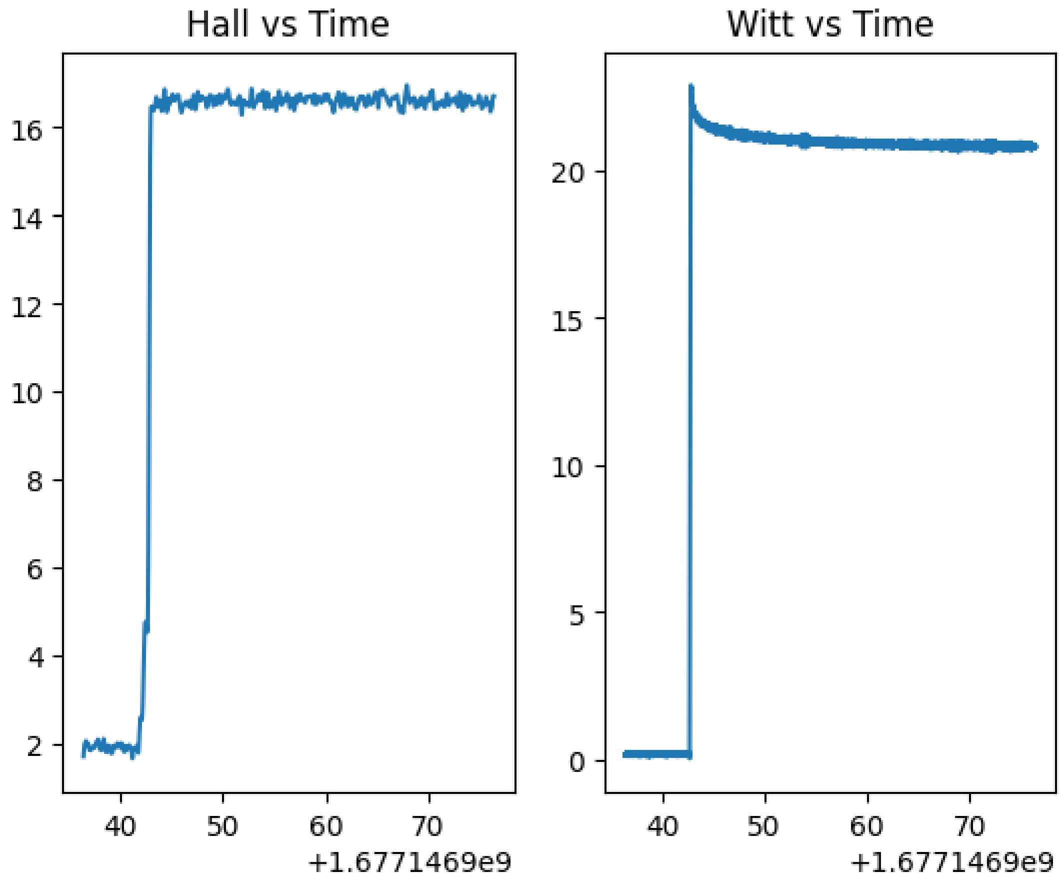
```
# hallRaw_df['hall'] = hall_filtered
# time = np.arange(0, len(hall_filtered), 1, dtype=int)
# hallRaw_df['time'] = time
```

In [53]:

```
figure, axis = plt.subplots(1,2)

axis[0].plot(time,hall_filtered)
axis[0].set_title("Hall vs Time")

# For Cosine Function
axis[1].plot(time, witt)
axis[1].set_title("Witt vs Time")
```

Out[53]:

```
Text(0.5, 1.0, 'Witt vs Time')
```

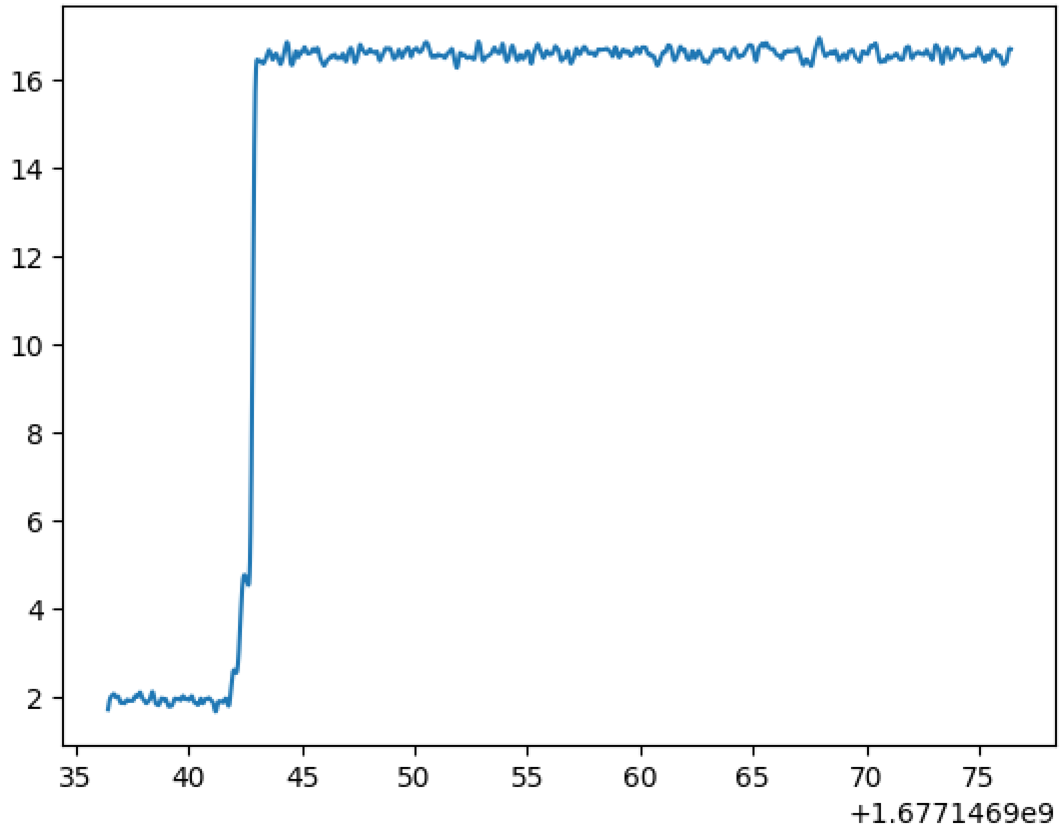


Savitzky-Golay Filter

```
In [75]: from scipy import signal

         hall_sgf = signal.savgol_filter(hall_filtered,
                                         550, # window size used for filtering
                                         2), # order of fitted polynomial
```

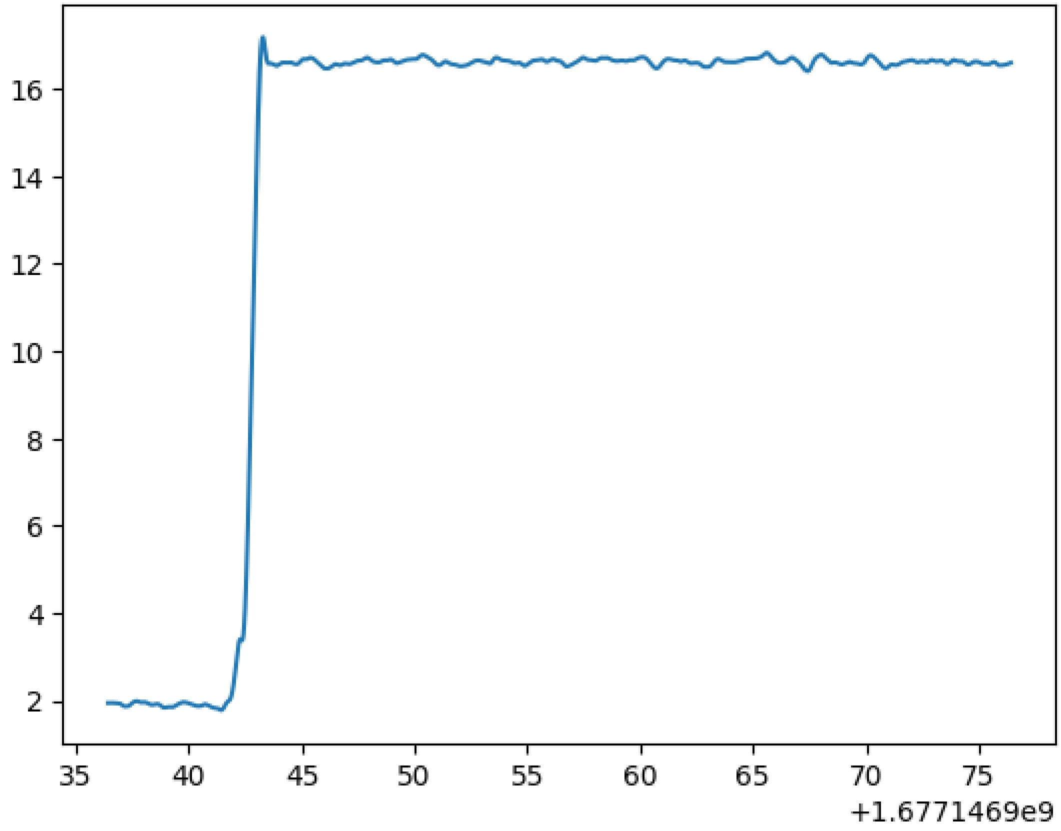
```
In [76]: plot(time, hall_filtered)
```

```
Out[76]: [<matplotlib.lines.Line2D at 0x7fbf292758d0>]
```



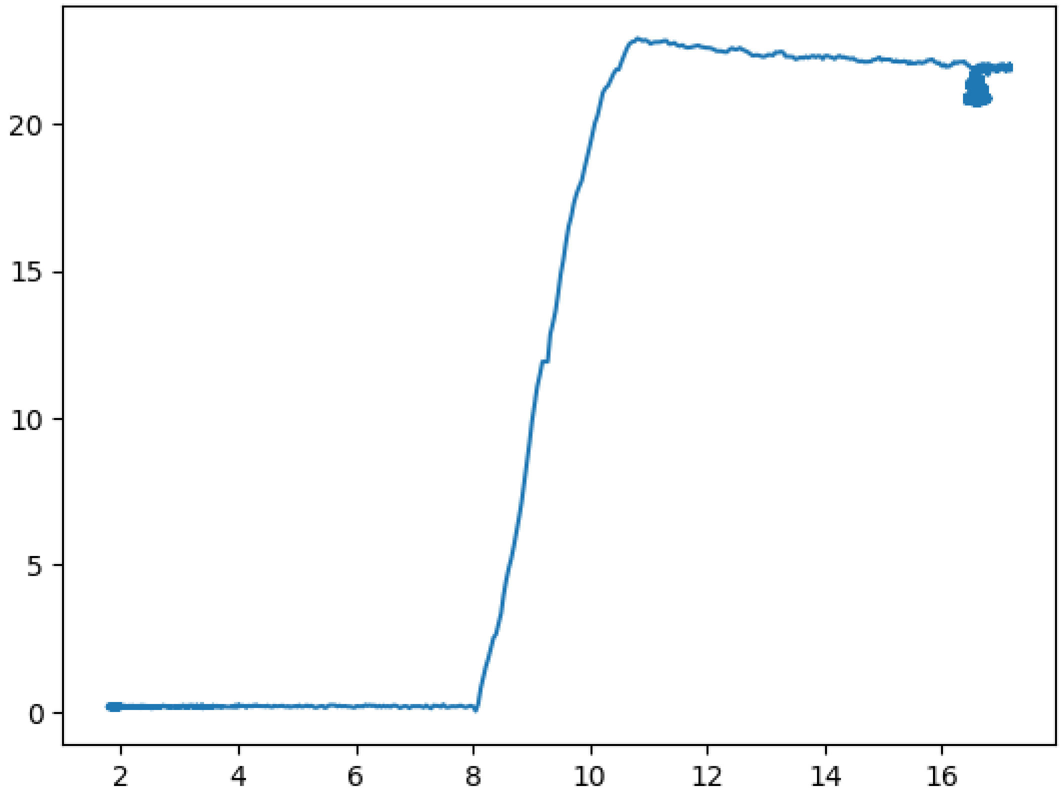
```
In [77]: plot(time, hall_sgf[0])
```

```
Out[77]: [<matplotlib.lines.Line2D at 0x7fbf290e68c0>]
```



```
In [78]: plot(hall_sgf[0],witt)
```

```
Out[78]: [<matplotlib.lines.Line2D at 0x7fbf29194220>]
```



```
In [ ]:
```