Neural signal acquisition & analysis: Julia coding workshop

Potas & Vickery

j.potas@unsw.edu.au

Analog vs digital

- What's the difference?
- Are any biological signals digital?
- Which one is better?

Recording an analogue signal introduces filtering, even without going through a filter unit.

Recording a digital signal introduces a host of sampling issues.

Filtering

Filtering is a critical issue in electrophysiology. The data is real-time, so make a mistake and it's gone!

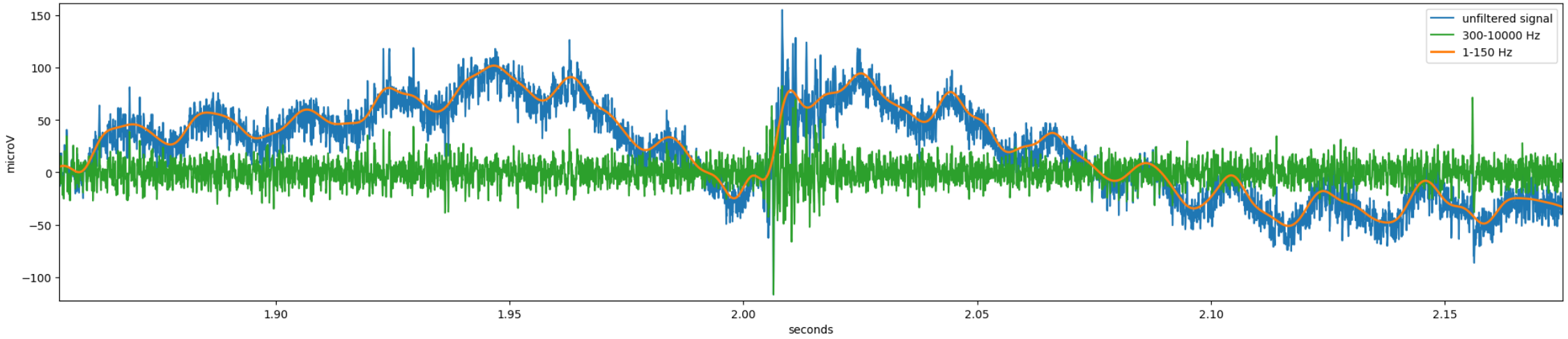
What is filtering?

Any time-varying signal is some combination of frequencies. A filter selectively excludes some range of frequencies. The most commonly used are:

- low pass (let all frequencies below X through)
- high pass (let all frequencies above Y through)
- bandpass (let frequencies between A & B through)

Why use filtering?

We often know the frequencies of signals we want to study. We can improve our “signal to noise ratio” if we shut out other frequencies.

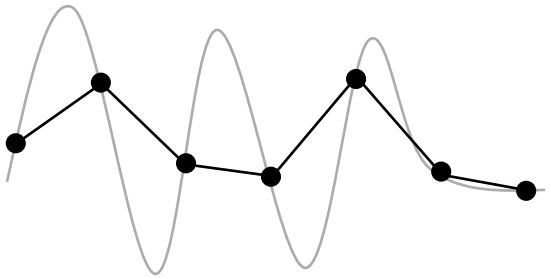


Data sampling

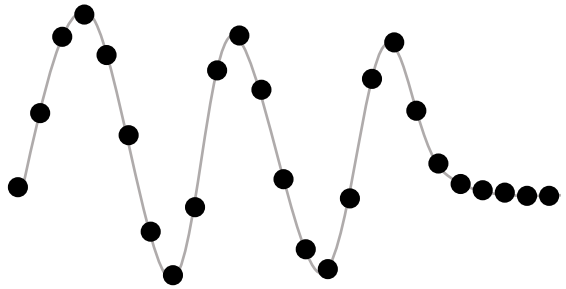
- When to sample? Fixed times, or variable to capture peaks?
- How fast to sample? Trade of the **Nyquist rate**, **aliasing** vs file size

Nyquist rate (i.e. minimum sampling rate) = 2x max frequency you want to capture

- Rule of thumb: sample at least 2x your low pass filter setting.
- How you choose the filter setting is a function of physiology and practicality.
- Consequences of under and over sampling:



7 data points
(not descriptive enough)



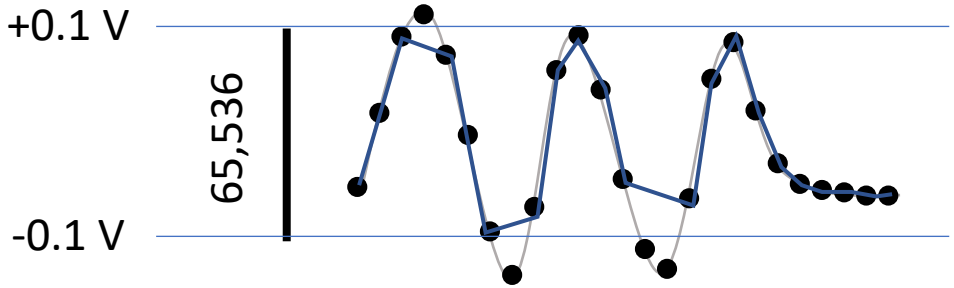
25 data points
(requires >3x more memory)

Data acquisition: key concepts

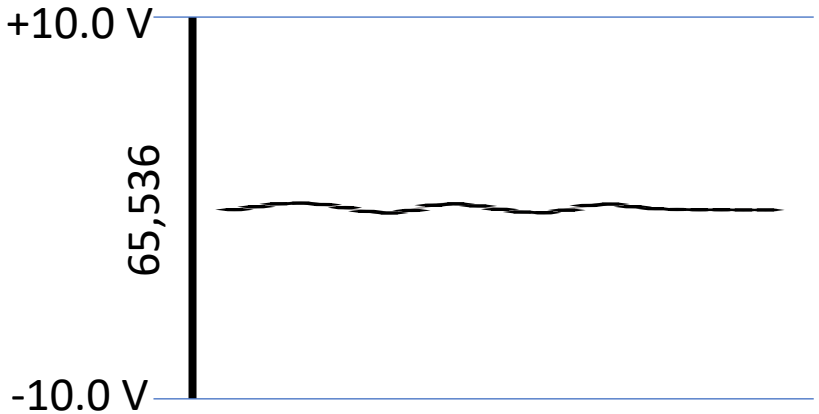
Digital resolution

- How many bits is your ADC?
- Data acquisition systems have a fixed resolution in the vertical range
  - The Powerlab system has 16 bit resolution: 2^{16} = 65,536
  - You need to define how may volts you wish to divide by 65,536
- **Voltage range**: consequences of inappropriate voltage range

- Use a pre-amp and/or amplifier to get the signal into the right range in terms of:
  - offset (i.e. base level from 0)
  - gain (i.e. amplification).



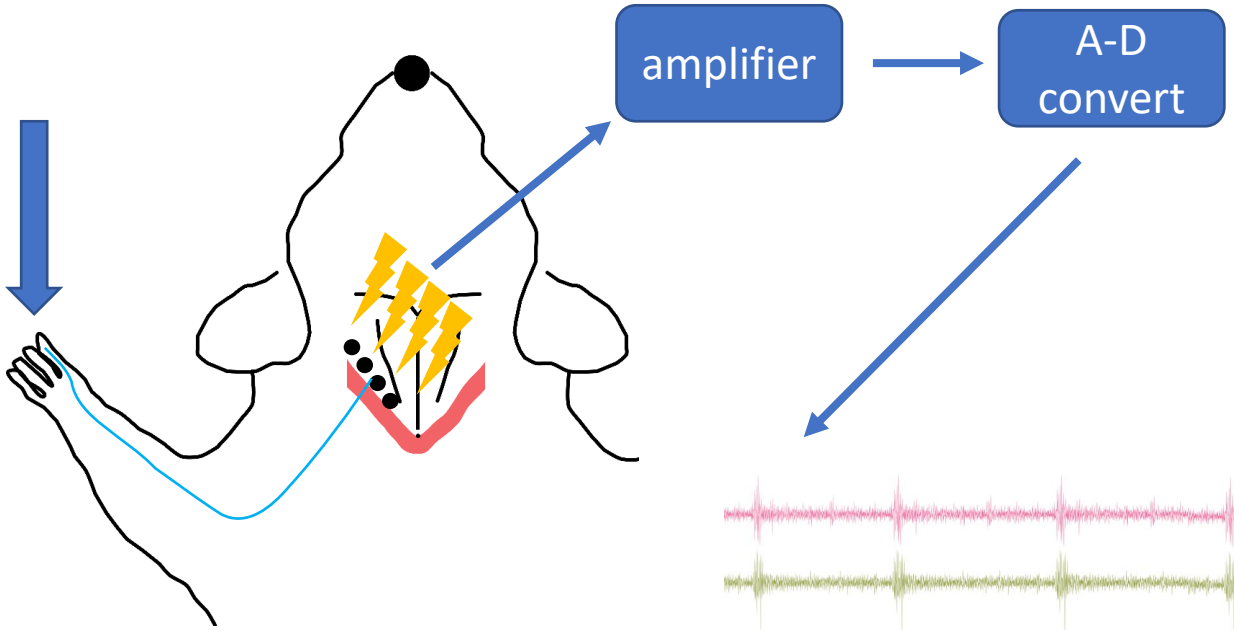
Too low voltage will give more vertical detail, but risks clipping the data

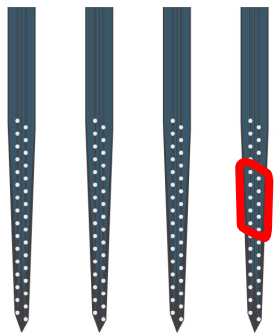


Too high voltage will capture everything, but you will have less vertical resolution

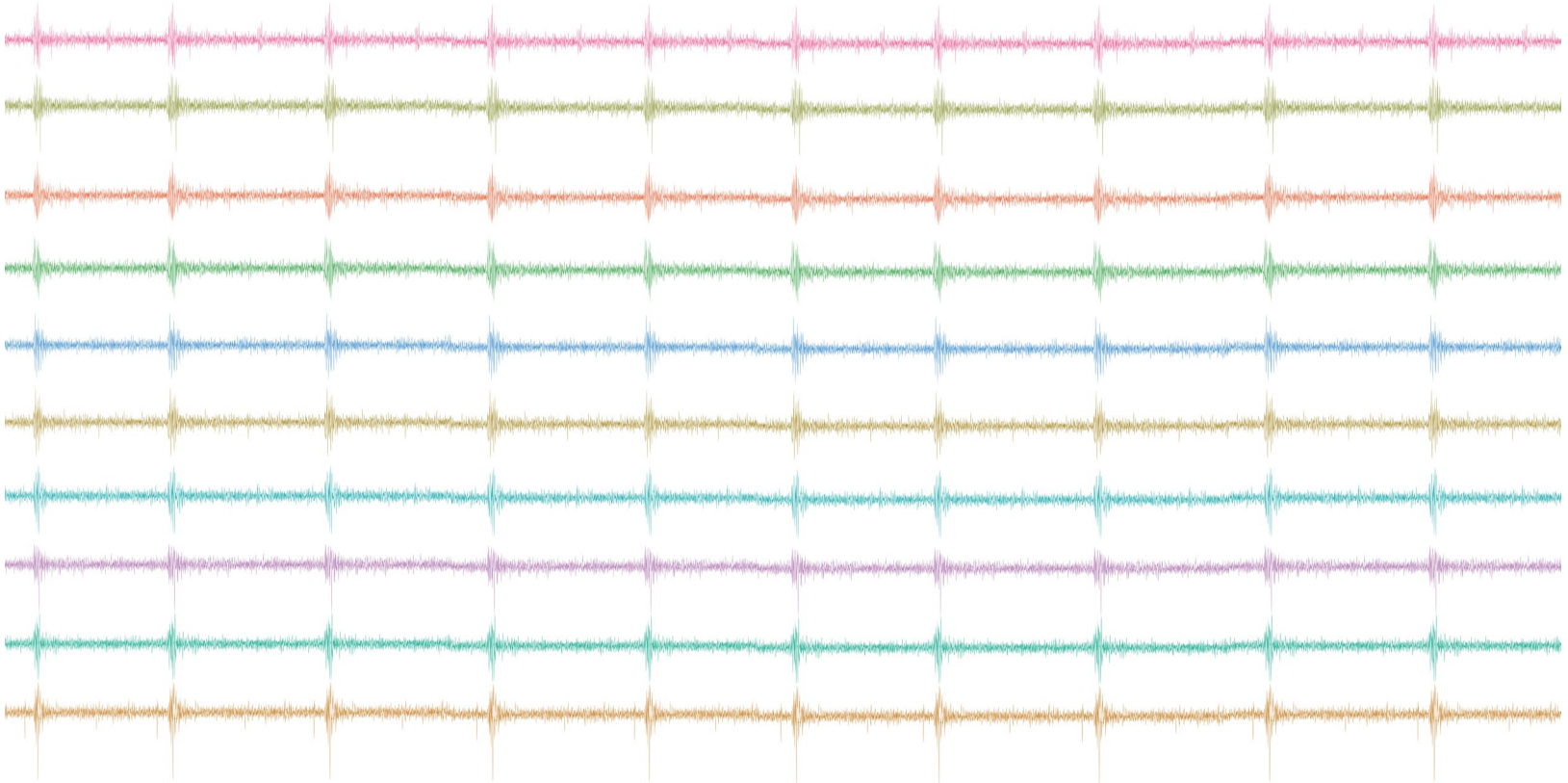
Experiment

- The following experiment explains how the data was acquired that you will import into Julia

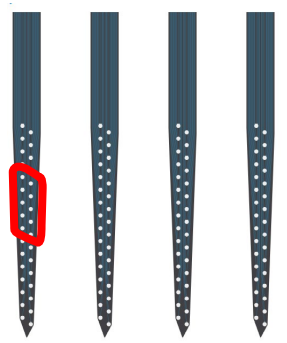




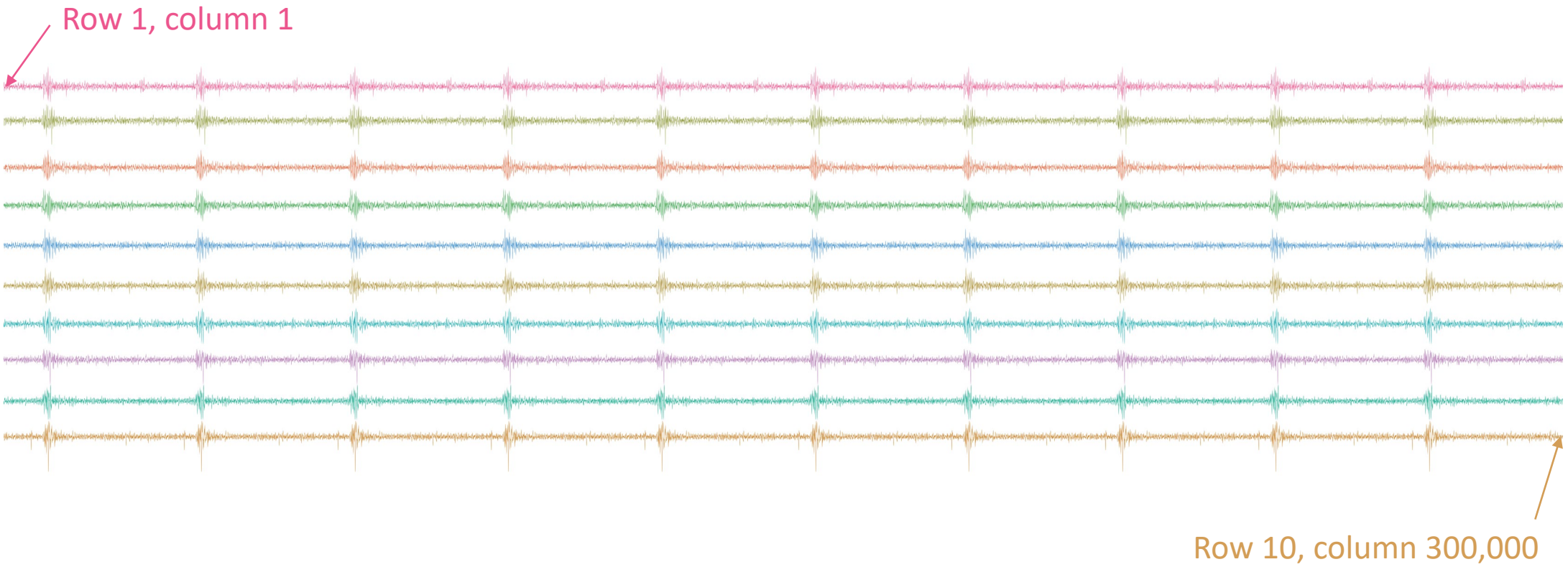
10 channels of data



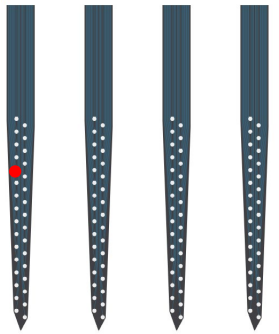
Coding for basic signal processing



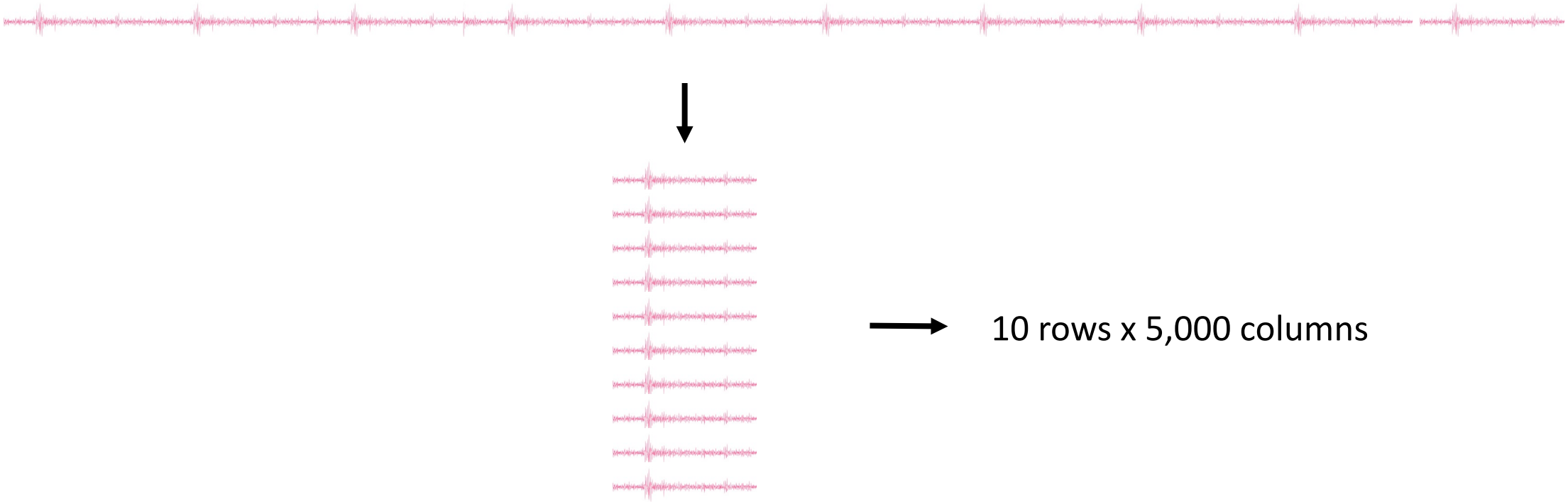
- We sample at 30 kHz
- Our data acquisition system produces a matrix, i.e. multiple vectors stacked on top of each other (typically 128 rows and 300,000 columns for a 10 s acquisition)
- Each row (vector) represents a single channel of data (e.g. showing 10 channels below)



Coding for basic signal processing



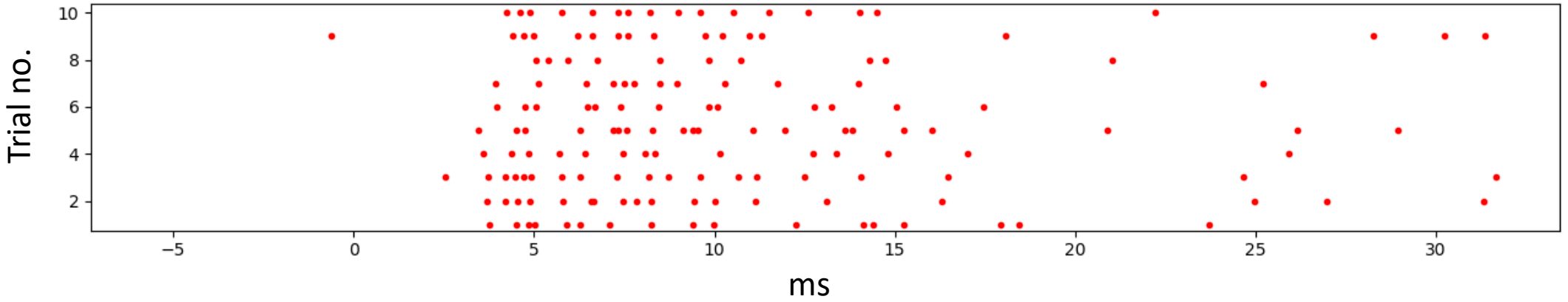
- Each channel contains 10 stimuli presented in sequence
  - a single vector (1 row x 300,000+ columns)
- We need to chop up the vector based on the stimulus and stack it into a matrix



Detecting peaks…

- Define a threshold (e.g. 3 x SD of some background/pre-stimulus region)
- Find peaks above that threshold and record their locations

- A raster plot shows detected events for multiple trials (each trial shown in a row):



Julia

- You need to install Julia and all the packages before class!! This is described from the **Installing Julia and Jupyter.pdf** document which is found on Moodle.
- Make sure you have uploaded the following two files from the Moodle page
    - neuro_data.jld2
    - JuliaWorkshop2020.ipynb
- Place these into a folder (something like “Julia Workshop 2020”)
- Ensure you can navigate to this folder!

Running a jupyter session of Julia:

- once you have opened a Jupyter Notebook session, navigate to the directory with the workshop files
- click on the file: JuliaWorkshop2020.ipynb. This should start up Julia inside a new browser tab.
- Each cell contains some code, which you can execute by hitting the "**|> Run**" button.

You can add cells by hitting the "+" button, so you can copy and paste code into it and modify it as you like. If you do that, I suggest you save the file as something else. To do that:

File -> save as -> add your new name without spaces after the "/".

- Run the 2nd block of code. Note that you should see an asterisk as follows, which means that Julia is thinking… Wait until the asterisk goes away before you execute the next cell:

In [*]:

- As we go through the explanation of the code in the workshop, you can make comments as you like in each code block. To do that, simply add a "#" before your type. You will see there are already many comments in the code to help explain key parts:

```
# anything after a "#" is not read by Julia, so you can write what you like. You will need to make a
# new one for each new line.
```

Julia tutorials:

- Those who are interested in learning more about Julia can find some great resources here: https://julialang.org/learning/
- If you are using PyPlot and can't find Julia documentation, you can look at the Python documentation, which explains the plotting functions reasonably well. The only thing you might need to remember is that Julia doesn't recognise 'single' quotes, only "double" quotes!