Tools for reproducible research in neuroscience

Yimeng Zhang
3rd year Ph.D. student in CS & CNBC
Lee Lab @ CMU
(http://leelab.cnbc.cmu.edu/)
March 07, 2016

Imbalance between data and tools

- New grants for brain research
  - MICrONS (http://microns.cnbc.cmu.edu/)
- Big data every day
  - Ca imaging, spiking, fMRI, EEG, MEG, …
- Need new tools to manage them.

Can you find the nobel prize in here?

Reproducibility in data analysis is a must

- help yourself rerun your work quickly in case your advisor wants you to build something new on top of it.

- produce legacy for the lab for future generation students.

- more collaboration among labs, and this would be painstaking without portable & reproducible code and data.

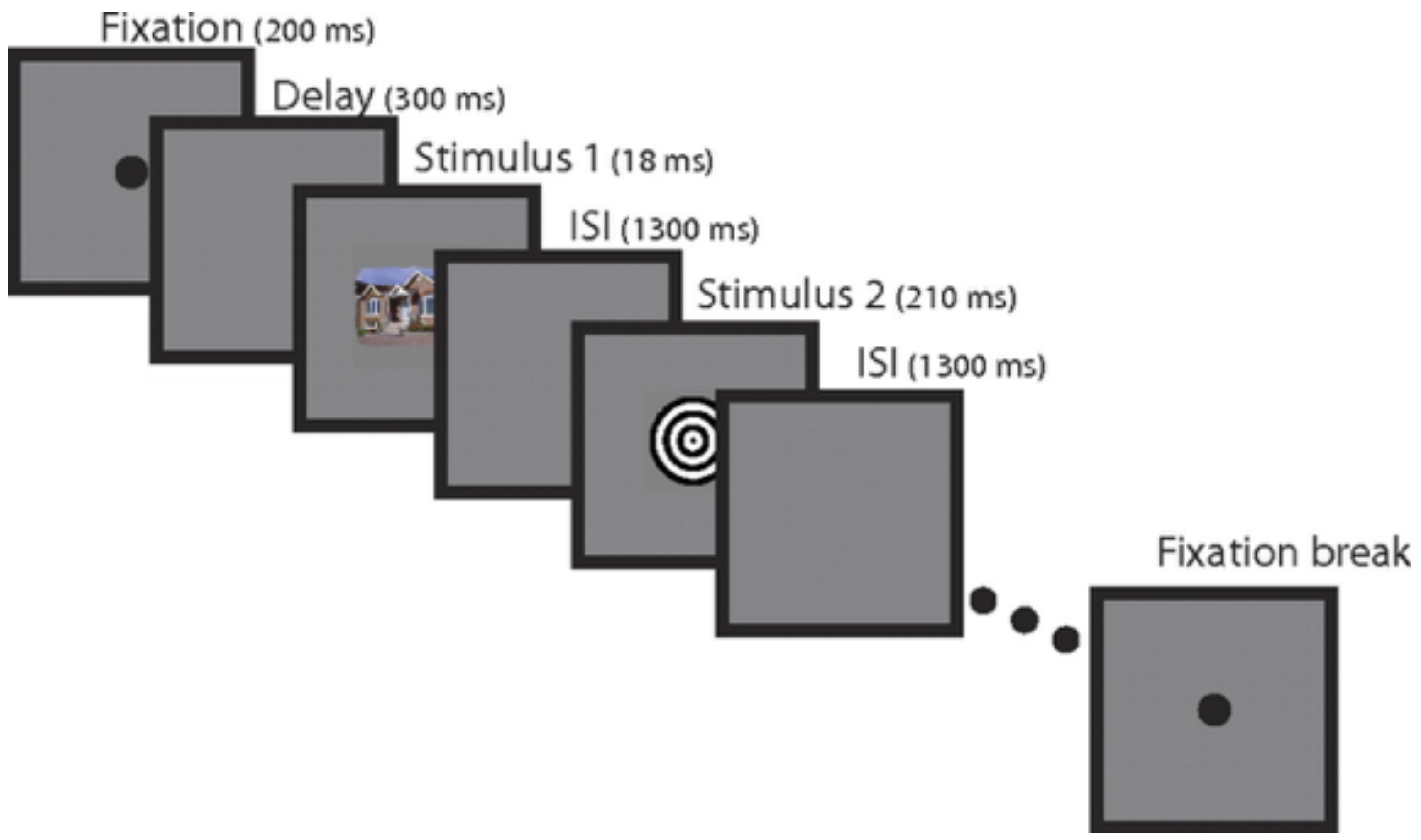
Tools for reproducible research in Lee Lab

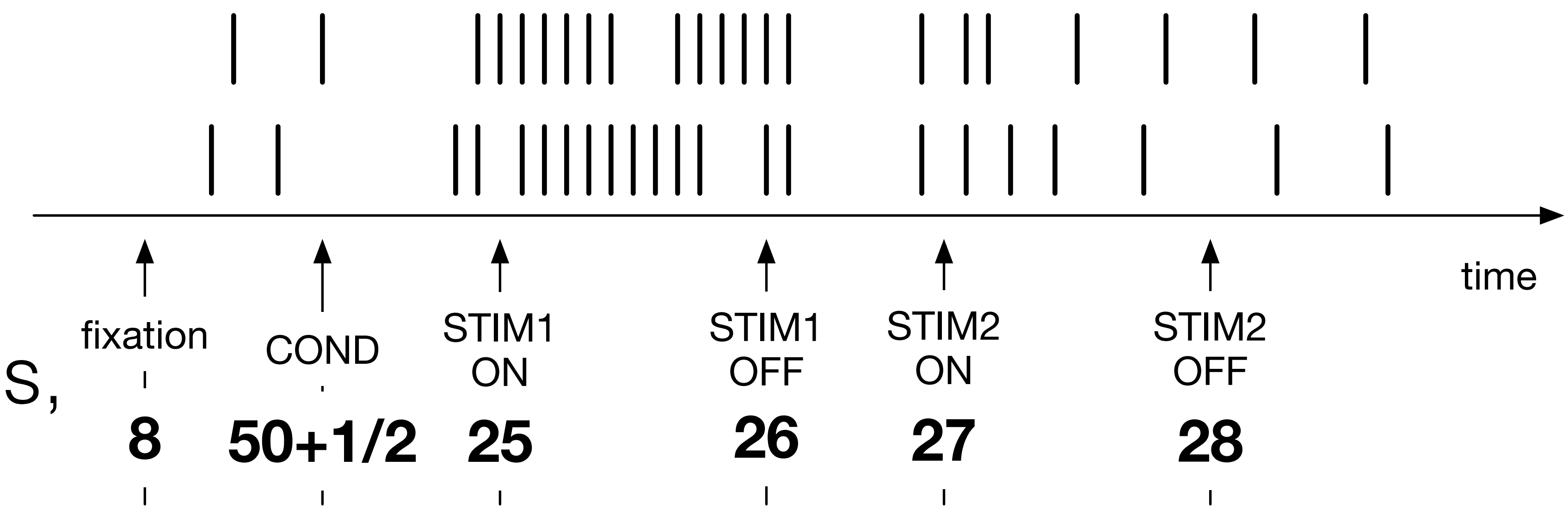
- **cdttable**: convert **trial-based** ~~spiking~~ **neural data** into a universal format (CDT table) for later analysis. (applicable to other trial-based data as well)

- **DataSMART**: data management and processing pipeline for science labs

typical experiment for many labs collecting data

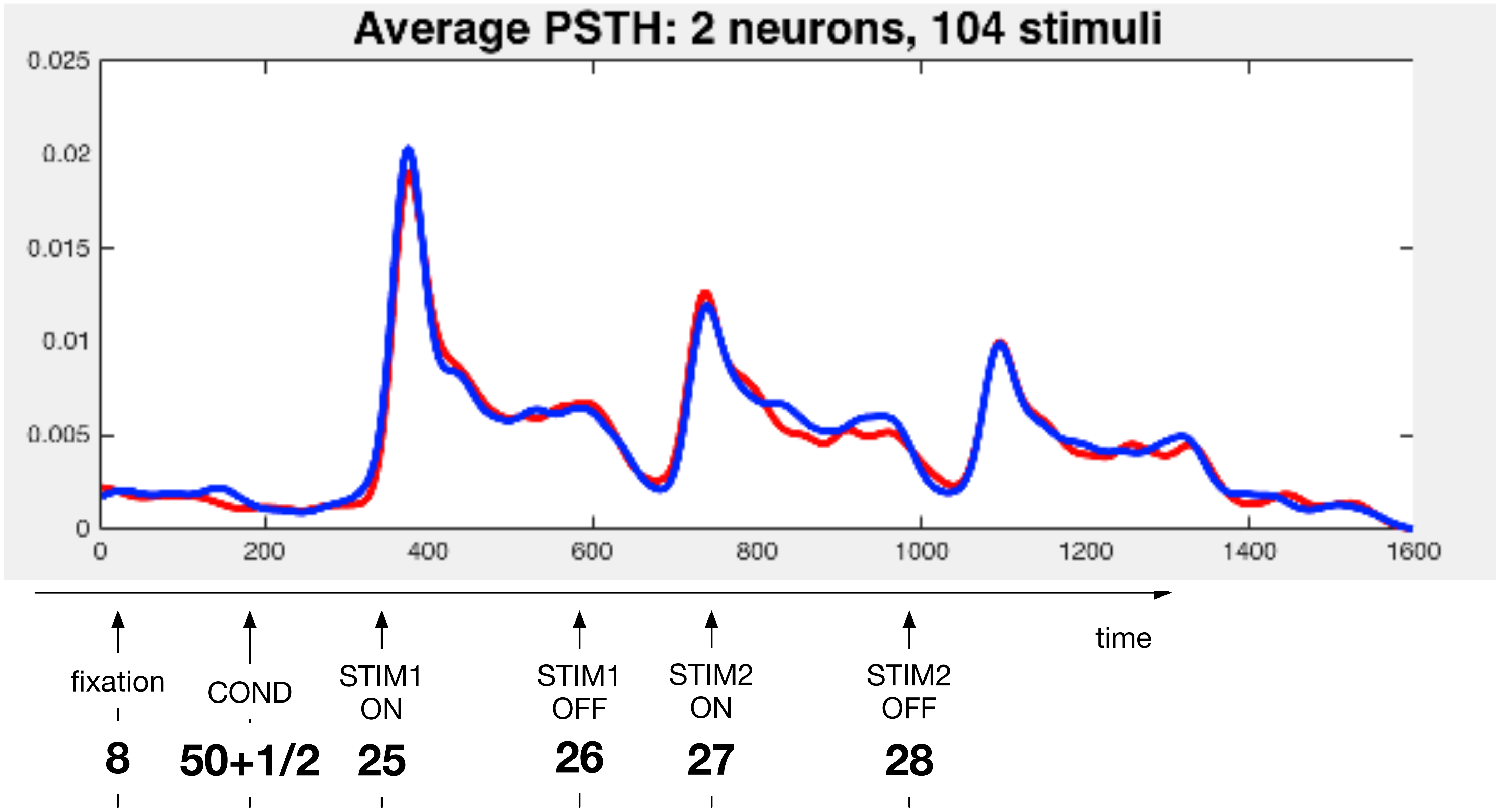
- a big `for` loop over trials
- each trial has a certain structure.
- events in the trial has **event codes**
- each trial has a **condition** (encoded as 50 + x)
- record spike times during the trial (two trials, one neuron shown)





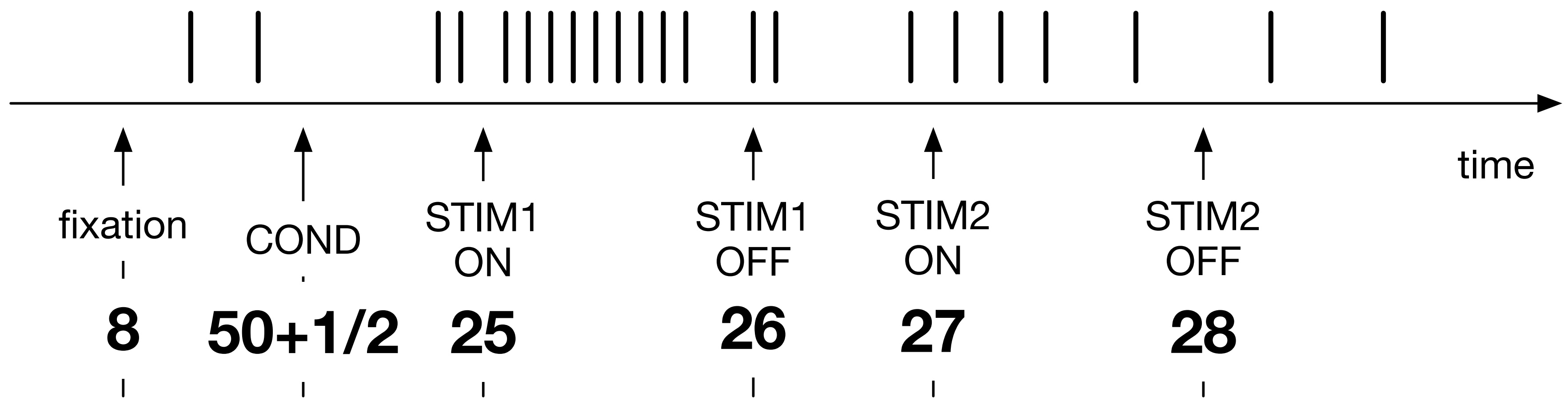
alignment is the next step

- align spiking data along certain codes in the trial, and plot PSTH, etc.
- alignment can be hard!



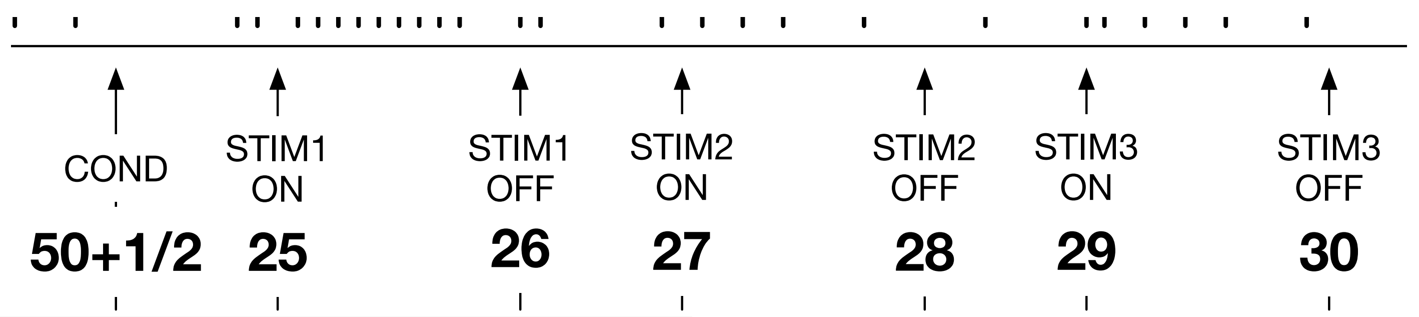
- For a specific task, alignment code is easy to write.
- But difficult to generalize across different experiments.
- Since these scripts are similar for different experiments, we tend to **copy & paste**

```
align_codes = [25,26,27,28]
[all_codes, all_times]=readNEV('20150321.nev');
for i = 1:nTrials
    codes = all_codes{i}; times = all_times{i};
    condition(i) = codes(2)-50;
    stim_1_start=times(codes==align_codes(1));
    stim_1_stop=times(codes==align_codes(2));
    stim_2_start=times(codes==align_codes(3));
    stim_2_stop=times(codes==align_codes(4));
end
```



`file 2`: copy & paste `file 1`

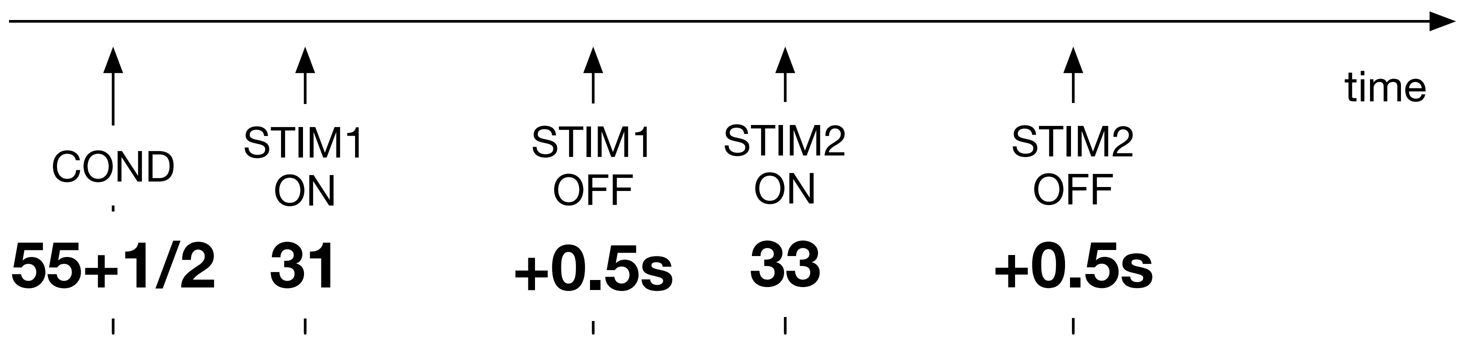
3 stimuli per trial



```
align_codes = [25,26,27,28,29,30]
[all_codes, all_times]=readNEV('20150421.nev');
for i = 1:nTrials
    codes = all_codes{i}; times = all_times{i};
    condition(i) = codes(2)-50;
    stim_1_start=times(codes==align_codes(1));
    stim_1_stop=times(codes==align_codes(2));
    stim_2_start=times(codes==align_codes(3));
    stim_2_stop=times(codes==align_codes(4));
    stim_3_start=times(codes==align_codes(5));
    stim_3_stop=times(codes==align_codes(6));
end
```

file 3: C & P file 2

trial end without marker



```
%align_codes = [25,26,27,28,29,30]
start_codes = [31,33]
[all_codes, all_times]=readNEV('20150521.nev');
for i = 1:nTrials
    codes = all_codes{i}; times = all_times{i};
    %condition(i) = codes(2)-50;
    condition(i) = codes(2)-55;
    stim_1_start=times(codes==start_codes(1));
    %stim_1_stop=times(codes==align_codes(2));
    stim_1_stop=stim_1_start + 0.5;
    stim_2_start=times(codes==start_codes(2));
    %stim_2_stop = times(codes==align_codes(4));
    stim_2_stop=stim_2_start+0.5;
end
```

Copy & paste is bad in the long run

- Redundancy of code — OK with big HD
- scripts will become longer and longer with `%commenting on and off%`, and **hacks specific to the experiment** would creep in.
- What if you **forget** removing these hacking lines for a good file in your next script?
- In the end, **unreliable** science and research.

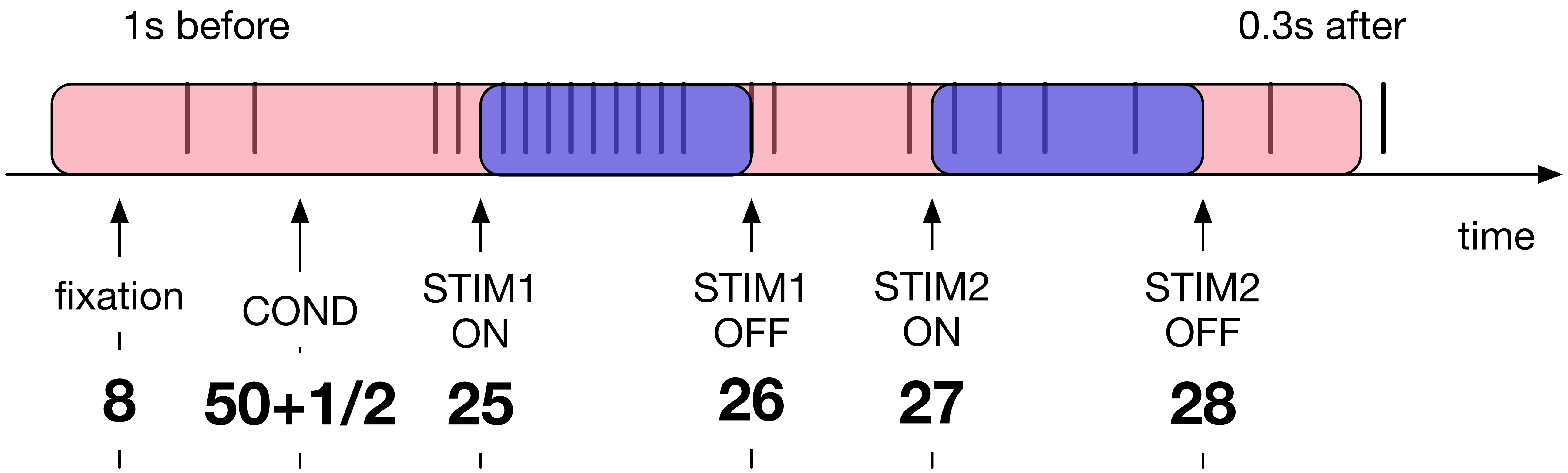
```
% file 1
[all_codes, all_times] = readNEV('faulty1.nev');
all_codes{378}{2}=0;
all_codes{887}{7}=0;

% file 2
[all_codes, all_times] = readNEV('faulty2.nev');
all_codes(107){19}=17;
all_codes(107){20}=101;
```

Solution

- Write a function for alignment and pass in a different set of parameters for each experiment.
- less redundancy, and easier for people to understand the code.
- discourage free-form hacking.
- **cdttable** is one solution.
- https://github.com/leelabcnbc/cdttable

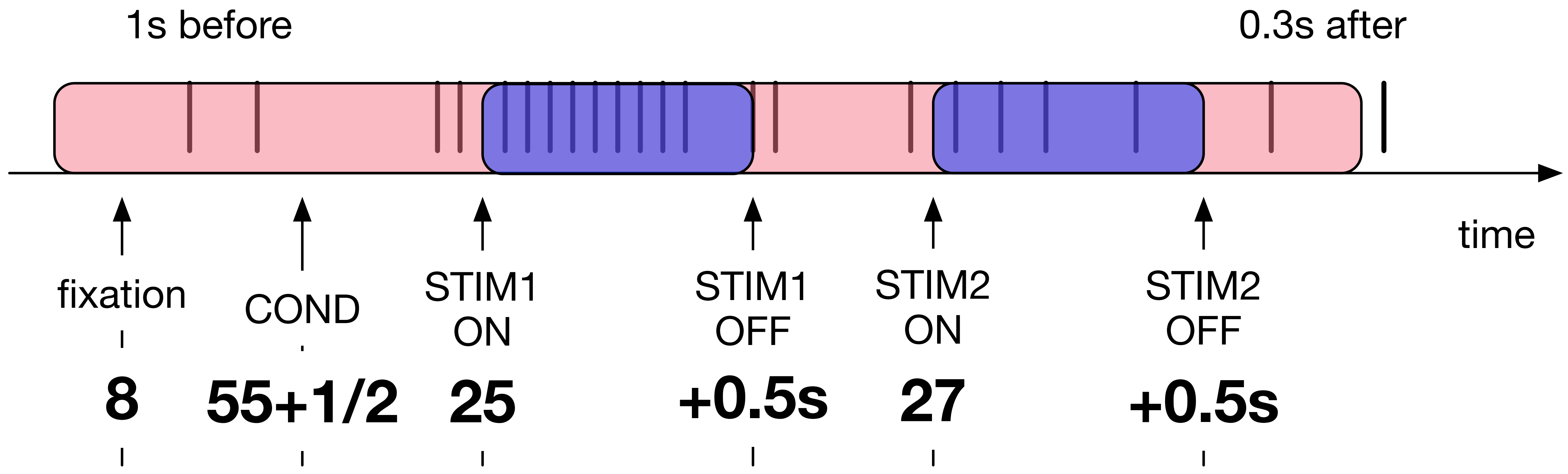
Parameter specification in **cdttable**

```
{
  "comment": "brain bag talk",
  "subtrials":[
    { "start_code":25, "end_code":26 },
    { "start_code":27, "end_code":28 },
  ],
  "margin_before":1.0, "margin_after":0.3,
  "trial_to_condition_func": "(x,idx) x(2)-50"
}
```



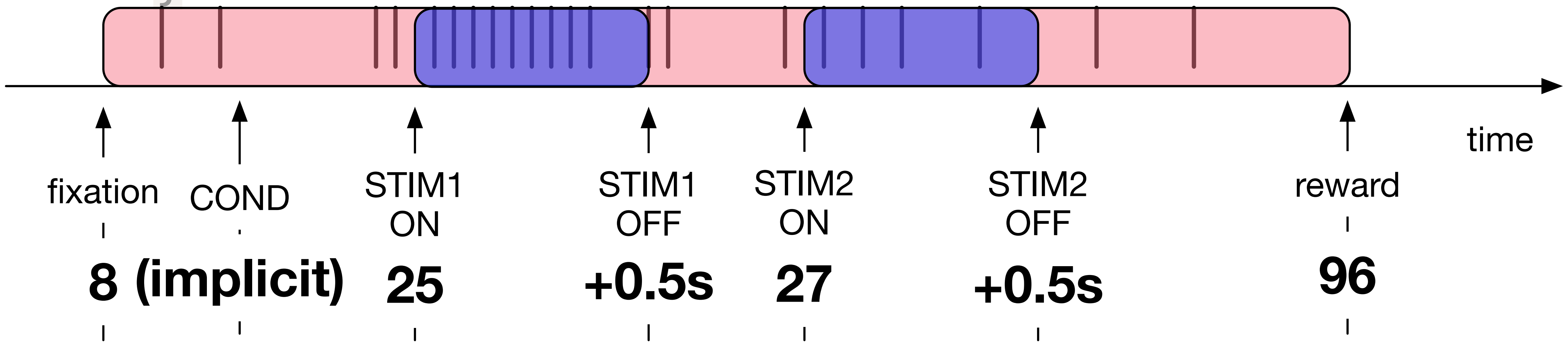
Parameter specification in **cdttable**

```
{
  "comment": "brain bag talk 2",
  "subtrials":[
    { "start_code":25, "end_time":0.5 },
    { "start_code":27, "end_time":0.5 },
  ],
  "margin_before":1.0, "margin_after":0.3,
  "trial_to_condition_func": "(x,idx) x(2)-55"
}
```



Parameter specification in **cdttable**

```
{
  "comment": "brain bag talk 3",
  "subtrials":[
    { "start_code":25, "end_time":0.5 },
    { "start_code":27, "end_time":0.5 },
  ],
  "trial_start_code":8, "trial_end_code":96,
  "margin_before":0, "margin_after":0,
  "trial_to_condition_func": "(x,idx) idx"
}
```



Demo

CDT table format

- CDT was the old format used in the lab.
    - No idea why called that.
- I make the CDT format more **tabular**, resulting in CDT table.
- Each trial is a row in the table, and each trial’s spikes a subtable
- with `cellfun`, many common operations can be done quickly.
    - Check docs online

condition	starttime	stoptime	event codes	event times	spike electrode	spike unit	spike times
1	0.0,0.51,0.99	0.4,0.81,1.39	25,26…	0.0,0.4,…	1	1	0.2,0.3,…
					1	2	0.1,0.2,…
					2	1	0.7,0.9,…
					3	1	0.8,1.0,…
					3	2	0.9,1.0…
					4	1	0.1,0.2…
					…	…	…

Questions?

- Code: https://github.com/leelabcnbc/cdttable
- Documentation: http://cdttable.readthedocs.org/
- Lab Website: http://leelab.cnbc.cmu.edu/