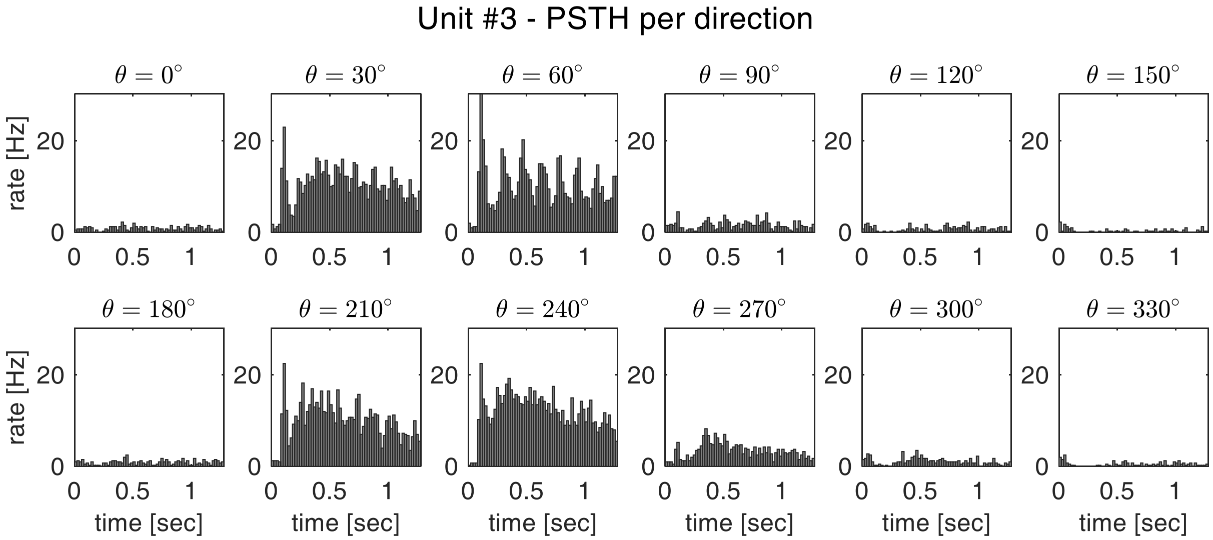
Exercise #2 – Analysis of orientation & direction selectivity in V1 cells

- Load the data file **`'SpikesX10U12D.mat.`** After the loading, you should see a structure array with the same name **`SpikesX10U12D`** in your workspace.
- This data represents an experiment using full field visual stimuli of moving periodic oriented gratings with the following features:
  - 12 directions of grating movement – evenly spaced on the circle ($\Delta\theta = 30^{o}$).
  - 200 repetitions (1.28 seconds duration) for each stimulus.
- The structure array contains simultaneous *in vivo* extra cellular recordings from 10 units (neurons) in the primary visual cortex (V1) of a monkey. It was recorded using a 10x10 electrode array that is permanently implanted in the monkey's V1. The given dataset is a chosen subset of all units extracted by a spike sorting procedure performed on the full 10x10 electrode recordings.
- The structure array itself has 3 dimensions (units, directions, repetition) and a single field (**`TimeList`**). Each **`TimeList`** cell/field contains a vector of spike events representing the activity of the chosen unit during the recorded repetition of the chosen direction
- Use **`SpikesX10U12D(i,j,k).TimeList`** to access each vector.

1. PSTH calculations and display

- *Methodological remark*: for understanding of the PSTH procedure refer to the previous Class exercise and its complementary slides.
- Create a 4 dimensional array for the PSTH data (i.e., with dimensions for the unit, the stimulus direction, the repetitions and the time bin of the PSTH.
- Use **`histogram`** or **`histcounts`** (fromer **`hist,  histc`**) to convert each **`TimeList`** into time binned vector.

- The duration of the time bin should be a parameter which is set at beginning, you should play with it until you are satisfied with the resulted figure.
- Choose a representative unit and display the 12 PSTH signals (for the 12 stimulus directions) on the same figure;
- Use **`subplot`** to plot the PSTH for each direction. Make sure to use the same scale of firing rates in all subplots.
- Use **`sgtitle`** to add a title for the whole figure (requires MATLAB R2018b or above).
- "A picture is worth a thousand words" – below is an example of the required output using unit 3 of the given dataset:



2. Orientation and direction tuning

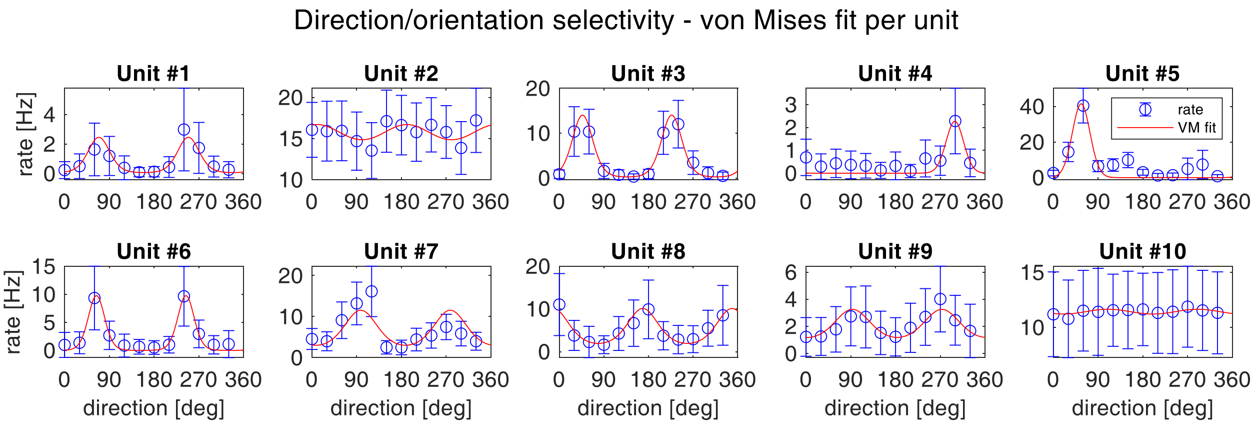
- **Note: To understand the fitting procedure and the von-Mises angular distribution function refer to the lesson slides.**
- For the purpose of this task, we shall use the total spike counts of each **TimeList** in the given array and ignore the temporal dimension of the recordings. To convert the total spike count into firing rate, divide it by the total duration of each trial.
- To estimate the conditional response and its error, use the **`mean`** and the **`std`** (standard deviation) functions over the different repetitions.

- For each unit, fit the mean response to a von-Mises angular distribution and plot the fitted curve on the appropriate figure panel.
- For each unit, try to fit the appropriate von Mises function for direction selectivity and orientation selectivity. In your plots, use the function that fits better for the specific unit (RMSE is lower). The goodness of fit is returned in the second output argument of the **`fit`** function, for further details please see MATLAB's **`help`** & **`doc`**.
- Fit the following function for each case:

```
% direction selective:
 VM_drct(x)  =  A * exp (k * cos (x - PO) );
% orientation selective:
 VM_ornt(x)  =  A * exp (k * cos (2*(x- PO)) );
% A, k & PO are fitted for independent variable x
```

  - In both functions **`A`**, **`k`** and **`PO`** are the fitted coefficients. **`x`** is the independent variable.

- Here are the tuning curves of the different units:



- For the required display use the **`subplot`**, **`plot`** and **`errorbar`** commands, e.g.:

```
figure ('Color', 'w', 'Units', 'centimeters', 'Position', [0 0 25 10]); hold on;
for unit_idx = 1:10
    subplot (2,5, unit_idx); hold on;  % create subplot on the main figure
     % plot the experimental data:
    errorbar (d, ResponseM, ResponseSD, 'o');
    % plot the fitted curve :
    plot (x_vec,  model(x_vec), 'r');
```

- Use vector **x_vec** to plot the fitted curve (i.e the function with the fitted coefficients), **x_vec** is should be defined in radians:

```
X_vec = 0:0.01:2*pi;
```

- Methodological remarks:
  - Use the **help** & **doc** command to learn about the **fit** and **fittype** functions and try to execute the example given in the slides.
  - Use **zeros**, **cell** or **nan** to allocate memory.
  - Note that the default trigonometric function accepts radians as input.
    - The vector **Directions** is defined in radians as**:**

      ```
      rad = deg*pi/180;
      ```
    - You can use **rad2deg** & **deg2rad** to convert radians to degrees and vice versa.
    - All plots should use only degrees, not radians.

3. Exercise deliverable

Deliverable should include the MATLAB code you used and two figures like the ones presented above. Submit a report according to the guidelines, summarizing your work.