Exploring the Experiment's Design

In this set of exercises, we'll get our first look at the experiment we'll be analyzing in this course; curated data from the Steinmetz et al, 2019 paper.

Today's data is focused on three CSV files, each containing sessions from a different stretch of data collection. They contain trial-level data from the experiment:

- steinmetz_winter2016.csv
- steinmetz_summer2017.csv
- steinmetz_winter2017.csv

Table of Contents

Loading and viewing data

Let's load in some csv data with MATLAB and see what we have!

Code	Description
`readtable('my_datafile.csv')`	read a csv file
`data(row_start:row:end, :)`	select rows from row_start to row_end from data
`height(my_table)`	count the number of rows in my_table
`width(my_table)`	count the number of columns in my_table
`my_table.Properties.VariableNames`	get the column names of my_table
`[table1;table2;table3]`	concatenate tables

Download datafiles

```
userpath(fullfile(fileparts(matlab.desktop.editor.getActiveFilename), "src"))
download_from_sciebo("https://uni-bonn.sciebo.de/s/QyHoxfrSF6JILQd", 'data/
steinmetz_winter2016.csv')
```

```
downloading file to data/steinmetz_winter2016.csv
Done!
```

```
download_from_sciebo("https://uni-bonn.sciebo.de/s/Z3QHxJztEueDQF8", 'data/
steinmetz_summer2017.csv')
```

```
downloading file to data/steinmetz_summer2017.csv
Done!
```

```
download_from_sciebo("https://uni-bonn.sciebo.de/s/9FxelLhARmHpw85", 'data/steinmetz_winter2017.csv')
```

```
downloading file to data/steinmetz_winter2017.csv
Done!
```

```
userpath(fullfile(fileparts(matlab.desktop.editor.getActiveFilename), "data"))
```

Example:

Load in the winter 2016 dataset and view the first 5 rows of the data

```
data_winter2016 = readtable('data/steinmetz_winter2016.csv');
data_winter2016(1:5,:)
```

```
ans = 5×15 table
```

...

	trial	active_trials	contrast_left	contrast_right	stim_onset
1	1	'True'	100	0	0.5000
2	2	'True'	0	50	0.5000
3	3	'True'	100	50	0.5000
4	4	'True'	0	0	0.5000
5	5	'True'	50	100	0.5000

The variable data is a MATLAB table, we can see that in the Workspace pannel on the right.

Exercises

Load in the summer 2017 dataset and view the first 10 rows of the data

```
data_summer2017 = readtable('data/steinmetz_summer2017.csv');
data_summer2017(1:10,:)
```

```
ans = 10×15 table
```

...

	trial	active_trials	contrast_left	contrast_right	stim_onset
1	1	'True'	0	50	0.5000
2	2	'True'	50	0	0.5000
3	3	'True'	0	0	0.5000
4	4	'True'	0	0	0.5000
5	5	'True'	0	0	0.5000

	trial	active_trials	contrast_left	contrast_right	stim_onset
6	6	'True'	0	0	0.5000
7	7	'True'	50	25	0.5000
8	8	'True'	0	25	0.5000
9	9	'True'	50	0	0.5000
10	10	'True'	0	50	0.5000

How many rows are in the summer 2017 dataset?

```
height(data_summer2017)
```

```
ans = 2747
```

How many columns are in the summer 2017 dataset?

```
width(data_summer2017)
```

```
ans = 15
```

What are the column names in the summer 2017 dataset?

```
data_summer2017.Properties.VariableNames
```

```
ans = 1×15 cell
'trial'     'active_trials''contrast_left''contrast_right''stim_onset''gocue_tim ···
```

Load in the winter 2017 dataset to an appropriately named variable

```
data_winter2017 = readtable('data/steinmetz_winter2017.csv');
```

How many rows are in the winter 2017 dataset?

```
height(data_winter2017)
```

```
ans = 7906
```

Combine the 3 datasets into one table named `data`

```
data = [data_winter2016; data_winter2017; data_summer2017];
```

How many trials are there in the combined dataset?

```
height(data)
```

```
ans = 14420
```

Are you sure that the combined dataset has all the rows of the 3 smaller datasets?

Count the rows in each to find out.

```
height(data_winter2017)+ height(data_summer2017)+ height(data_winter2016) ==
height(data)
```

```
ans = logical
   1
```

Experiment Description: Calculating Statistics on Continuous Data

Let's calculate some simple statistics from the dataset.

Code	Description
`my_table.column_A`	access column_A of my_table
`min(my_data)`	find the minimum value of my_data
`max(my_data)`	find the maximum value of my_data
`mean(my_data)`	find the mean value of my_data
`mean(my_data, "omitmissing")`	calculate the mean ignoring missing values
`median(my_data)`	find the media value of my_data
`std(my_data)`	find the standard deviation of my_data
`range(my_data)`	find the difference between largest and smallest values in my_data

Example

Find the minimum response time in the data

```
min(data.response_time)
```

```
ans = 0.4794
```

Exercises

Find the maximum response time in the data

```
max(data.response_time)
```

```
ans = 2.7136
```

Find the maximum gocue time in the data

```
max(data.gocue_time)
```

```
ans = 1.1988
```

Find the minimum gocue time in the data

```
min(data.gocue_time)
```

```
ans = 0.3951
```

Find the mean response time in the data. **Hint** - as the data contains missing values use the "omitmissing" option.

```
mean(data.response_time, "omitmissing")
```

```
ans = 1.4228
```

Calculate the mean of response time including missing values. How do missing values affect the result?

```
mean(data.response_time)
```

```
ans = NaN
```

Find the median response time in the data

```
median(data.response_time, "omitmissing")
```

```
ans = 1.1833
```

Find the standard deviation of response time

```
std(data.response_time,"omitmissing")
```

```
ans = 0.6614
```

Find the standard deviation of gocue time

```
std(data.gocue_time,"omitmissing")
```

```
ans = 0.2005
```

Find the range of gocue time

```
range(data.gocue_time)
```

```
ans = 0.8038
```

Find the range of feedback time using the `max()` and `min()` functions

```
max(data.feedback_time) - min(data.feedback_time)
```

```
ans = 2.2438
```

Experiment Description: Calculating Statistics of Across Categorical Data

In data science we often want to perform analysis on distinct separate categories, for example, analysing trials that occured on different days.

To do this, we group the data and analyse each group separately.

Code	Description
`groupsummary(data, "column_A")`	Group data according to column_A and count occurrances
`groupsummary(data, "column_A", "mean", "column_B")`	Group data according to column_A and calculate the mean of column_B
`groupsummary(data, ["column_A", "column_C"])`	Group data according to both column_A and column_C and count occurances
`max(data.column_A)`	Compute the maximum value in column_A of data
`mean(data.column_A)`	Compute the mean values of column_A in data

Example Exercise

How many trials occurred for each session date?

```
groupsummary(data,"session_date")
```

```
ans = 31×2 table
```

	session_date	GroupCount
1	14-Dec-2016	364
2	17-Dec-2016	401
3	18-Dec-2016	378
4	07-Jan-2017	554
5	08-Jan-2017	632
6	09-Jan-2017	585
7	10-Jan-2017	363
8	11-Jan-2017	252
9	12-Jan-2017	238
10	15-May-2017	357

	session_date	GroupCount
11	16-May-2017	345
12	18-May-2017	194
13	15-Jun-2017	360
14	16-Jun-2017	482

⋮

Exercises

How many trials did each mouse participate in?

```
groupsummary(data,"mouse")
```

ans = 10×2 table

	mouse	GroupCount
1	'Cori'	1143
2	'Forssmann'	1485
3	'Hench'	1851
4	'Lederberg'	2902
5	'Moniz'	896
6	'Muller'	1112
7	'Radnitz'	1512
8	'Richards'	1677
9	'Tatum'	1389
10	'Theiler'	453

What was the mean response time for each mouse?

```
groupsummary(data,"mouse", "mean", "response_time")
```

ans = 10×3 table

	mouse	GroupCount	mean_response_time
1	'Cori'	1143	1.5909
2	'Forssmann'	1485	1.4706
3	'Hench'	1851	1.5257
4	'Lederberg'	2902	1.1438
5	'Moniz'	896	1.7777
6	'Muller'	1112	1.5758
7	'Radnitz'	1512	1.5591

	mouse	GroupCount	mean_response_time
8	'Richards'	1677	1.3583
9	'Tatum'	1389	1.3377
10	'Theiler'	453	1.3388

What was the minimum response time for each mouse?

```
groupsummary(data,"mouse", "min", "response_time")
```

ans = 10×3 table

	mouse	GroupCount	min_response_time
1	'Cori'	1143	0.5682
2	'Forssmann'	1485	0.4949
3	'Hench'	1851	0.5522
4	'Lederberg'	2902	0.4794
5	'Moniz'	896	0.6826
6	'Muller'	1112	0.5854
7	'Radnitz'	1512	0.5838
8	'Richards'	1677	0.5059
9	'Tatum'	1389	0.5136
10	'Theiler'	453	0.4826

What was the most common (ie. mode) reaction_type for each mouse?

```
groupsummary(data,"mouse", "mode", "reaction_type")
```

ans = 10×3 table

	mouse	GroupCount	mode_reaction_type
1	'Cori'	1143	1
2	'Forssmann'	1485	-1
3	'Hench'	1851	1
4	'Lederberg'	2902	-1
5	'Moniz'	896	-1
6	'Muller'	1112	1
7	'Radnitz'	1512	-1
8	'Richards'	1677	0
9	'Tatum'	1389	1
10	'Theiler'	453	-1

What was the range of response times for each session date?

```
groupsummary(data,"session_date", "range", "response_time")
```

ans = 31×3 table

	session_date	GroupCount	range_response_time
1	14-Dec-2016	364	2.1054
2	17-Dec-2016	401	2.1227
3	18-Dec-2016	378	2.1076
4	07-Jan-2017	554	2.1245
5	08-Jan-2017	632	2.1158
6	09-Jan-2017	585	2.0917
7	10-Jan-2017	363	2.0796
8	11-Jan-2017	252	2.0306
9	12-Jan-2017	238	1.9787
10	15-May-2017	357	1.9979
11	16-May-2017	345	1.9422
12	18-May-2017	194	2.0036
13	15-Jun-2017	360	2.1031
14	16-Jun-2017	482	2.1031

⋮

Rerun the last exercise replaceing "range" with "all"

```
groupsummary(data,"session_date", "all", "response_time")
```

ans = 31×14 table

...

	session_date	GroupCount	mean_response_time	sum_response_time
1	14-Dec-2016	364	1.5891	340.0590
2	17-Dec-2016	401	1.6215	407.0063
3	18-Dec-2016	378	1.5588	355.4048
4	07-Jan-2017	554	1.4704	652.8560
5	08-Jan-2017	632	1.5457	636.8091
6	09-Jan-2017	585	1.5928	581.3877
7	10-Jan-2017	363	1.5594	394.5252
8	11-Jan-2017	252	1.7116	243.0486
9	12-Jan-2017	238	1.7461	223.5022

	session_date	GroupCount	mean_response_time	sum_response_time
10	15-May-2017	357	1.8110	447.3048
11	16-May-2017	345	1.7360	407.9650
12	18-May-2017	194	1.7903	221.9923
13	15-Jun-2017	360	1.5966	399.1487
14	16-Jun-2017	482	1.5267	567.9217

⋮

What was the largest number of trials that any mouse took part in? **Hint** - the `groupsummary` function returns a table, from which you can perform compuations on

```
mouse_trial_count = groupsummary(data,"mouse");
max(mouse_trial_count.GroupCount)
```

```
ans = 2902
```

How many trials did each mouse take part in on each session date?

```
groupsummary(data,["mouse","session_date"])
```

ans = 39×3 table

	mouse	session_date	GroupCount
1	'Cori'	14-Dec-2016	364
2	'Cori'	17-Dec-2016	401
3	'Cori'	18-Dec-2016	378
4	'Forssmann'	01-Nov-2017	359
5	'Forssmann'	02-Nov-2017	364
6	'Forssmann'	04-Nov-2017	400
7	'Forssmann'	05-Nov-2017	362
8	'Hench'	15-Jun-2017	360
9	'Hench'	16-Jun-2017	482
10	'Hench'	17-Jun-2017	557
11	'Hench'	18-Jun-2017	452
12	'Lederberg'	05-Dec-2017	450
13	'Lederberg'	06-Dec-2017	410
14	'Lederberg'	07-Dec-2017	378

⋮

What was the average number of trials that each mouse participated in in a session?

```
mouse_session_count = groupsummary(data,["mouse","session_date"]);
mean(mouse_session_count.GroupCount)
```

```
ans = 369.7436
```

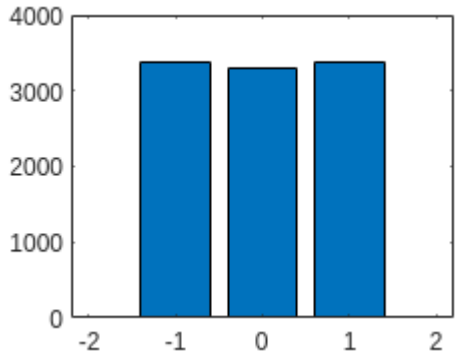
Visualizing the Experimental Design using Plots

Code	Description
`bar(x_data, y_data)`	make a bar plot of x and y data
`bar( x_data, y_data, "green")`	make a bar plot with green bars
`sortrows(table, "column_A")`	sort a table according to values in column_A
`ylabel("a new label")`	set the y label of a plot
`categorical(data_values)`	convert data_values to be categorical
`piechart(categorical_data)`	make a piechart from categorical_data
`swarmchart(categorical_data, y_data)`	make a swarmchart from categorical data and numerical y_data

Example Exercise

Make a bar plot showing how many times each response type appeared in the data

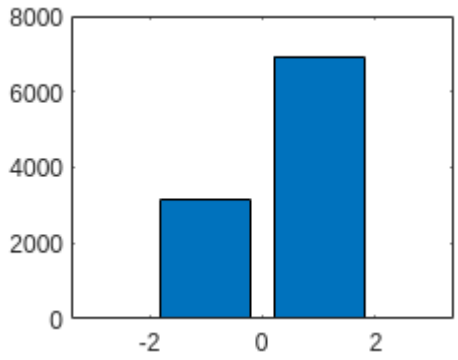
```
response_type_count = groupsummary(data, "response_type");
bar(response_type_count.response_type, response_type_count.GroupCount)
```



Exercises

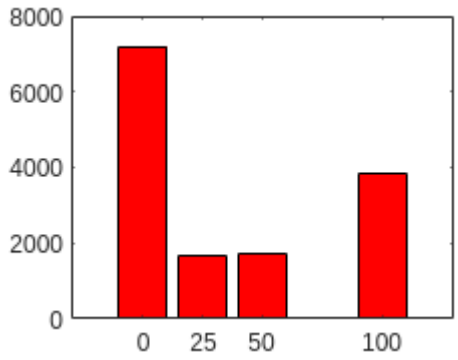
Plot the counts of feedback type from the dataset as a bar plot

```
response_type_count = groupsummary(data, "feedback_type");
bar(response_type_count.feedback_type, response_type_count.GroupCount)
```



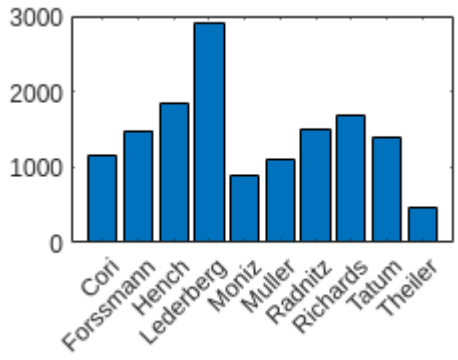
Make a bar plot showing the frequency of each `contrast_left` value with red bars

```
contrast_left_count = groupsummary(data, "contrast_left");
bar(contrast_left_count.contrast_left, contrast_left_count.GroupCount, 'red')
```



Make a bar plot showing how many trials each mouse participated in

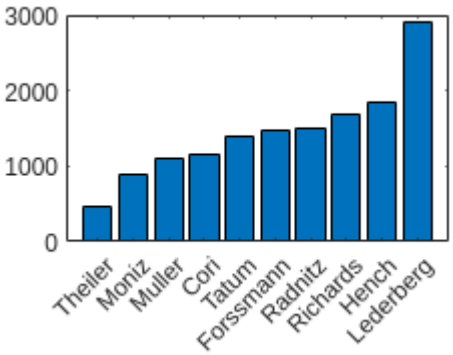
```
trials_by_mouse = groupsummary(data, "mouse");
bar(trials_by_mouse.mouse, trials_by_mouse.GroupCount)
```



Recreate the above bar plot, but sorting the bars from smallest to largest.

```
trials_by_mouse = groupsummary(data, "mouse");
trials_by_mouse = sortrows(trials_by_mouse, "GroupCount");
```

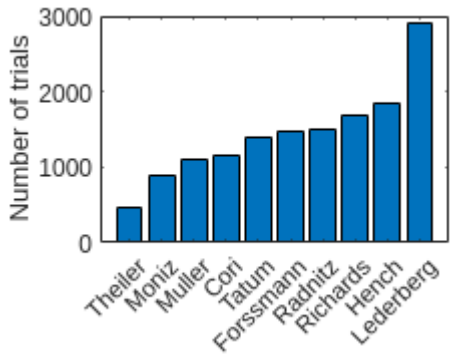
```
bar(trials_by_mouse.mouse, trials_by_mouse.GroupCount)
```



Add an appropriate y label to your plot

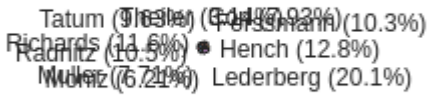
```
trials_by_mouse = groupsummary(data, "mouse");
trials_by_mouse = sortrows(trials_by_mouse, "GroupCount");

bar(trials_by_mouse.mouse, trials_by_mouse.GroupCount)
ylabel('Number of trials')
```



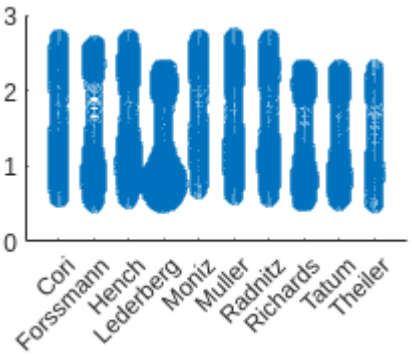
Make a pie chart showing the percentage of trials that each mouse participated in. **Hint** - the input to piechart must be categorical data.

```
mice = categorical(data.mouse);
piechart(mice)
```



Make a swarmchart showing the response time distributions for each mouse. **Hint** - the x data inputted to swarmchart must be categorical data

```
swarmchart(categorical(data.mouse), data.response_time)
```



Constructing a 3D bar plot

The following code makes a 3D bar plot showing how many trials each mouse underwent on each session date for the summer 2017 dataset.

You can see that several lines are "hard-coded", like the `session_dates` and y tick labels. Your challenge is to remove this hard-coding to make the plot generation generalisable.

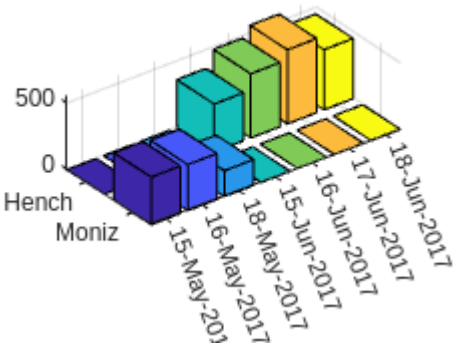
Code	**Description**
`unique(my_array)`	find the unique values in `my_array`
`length(my_array)`	find the number of elements in `my_array`
`reshape(my_matrix, n_cols, n_rows)`	reshape a matrix to have `n_cols` columns and `n_rows` rows
`xticks([start: interval: end])`	put xticks at x values defined by `start`, `interval` and `end`

```
mouse_names = ["Hench", "Moniz"];
```

```
session_dates = ["15-May-2017","16-May-2017", "18-May-2017",...
    "15-Jun-2017","16-Jun-2017", "17-Jun-2017", "18-Jun-2017"];

mouse_session_counts=groupcounts({data_summer2017.mouse,
data_summer2017.session_date}, "IncludeEmptyGroups",true);


reshaped_data_for_plot = reshape(mouse_session_counts,7,2)';
bar3(reshaped_data_for_plot)
yticklabels(["Hench","Moniz"])
xticks([1:1: 7])
xticklabels(datestr(session_dates))
```

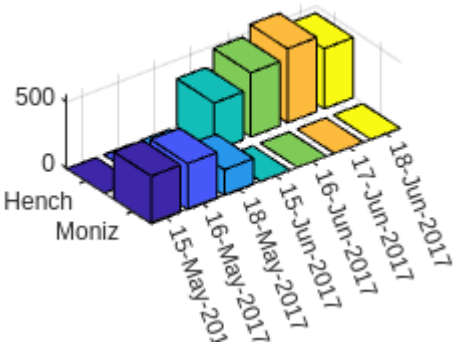


Remake the above plot, removing the hard-coding from the variable `mouse_names`

```
mouse_names = unique(data_summer2017.mouse);
session_dates = ["15-May-2017","16-May-2017", "18-May-2017",...
    "15-Jun-2017","16-Jun-2017", "17-Jun-2017", "18-Jun-2017"];

mouse_session_counts=groupcounts({data_summer2017.mouse,
data_summer2017.session_date}, "IncludeEmptyGroups",true);


reshaped_data_for_plot = reshape(mouse_session_counts,7,2)';
bar3(reshaped_data_for_plot)
yticklabels(["Hench","Moniz"])
xticks([1:1: 7])
xticklabels(datestr(session_dates))
```

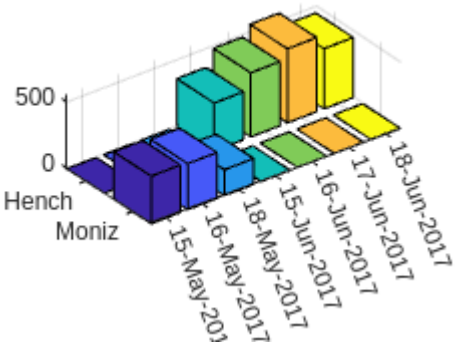


Remake the above plot, removing the hard-coding from the variable `session_dates`

```
mouse_names =unique(data_summer2017.mouse);
session_dates = unique(data_summer2017.session_date);

mouse_session_counts=groupcounts({data_summer2017.mouse,
data_summer2017.session_date}, "IncludeEmptyGroups",true);


reshaped_data_for_plot = reshape(mouse_session_counts,7,2)';
bar3(reshaped_data_for_plot)
yticklabels(["Hench","Moniz"])
xticks([1:1: 7])
xticklabels(datestr(session_dates))
```

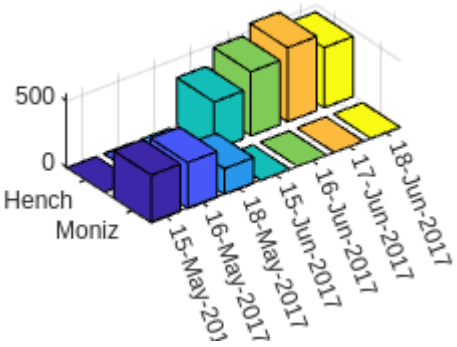


The `reshape` that produces `reshaped_data_for_plot` contains hard-coded numbers. Replace these with appropriate functions

```
mouse_names =unique(data_summer2017.mouse);
session_dates = unique(data_summer2017.session_date);

mouse_session_counts=groupcounts({data_summer2017.mouse,
data_summer2017.session_date}, "IncludeEmptyGroups",true);
```

```
reshaped_data_for_plot =
reshape(mouse_session_counts,length(session_dates),length(mouse_names))';
bar3(reshaped_data_for_plot)
yticklabels(["Hench","Moniz"])
xticks([1:1: 7])
xticklabels(datestr(session_dates))
```

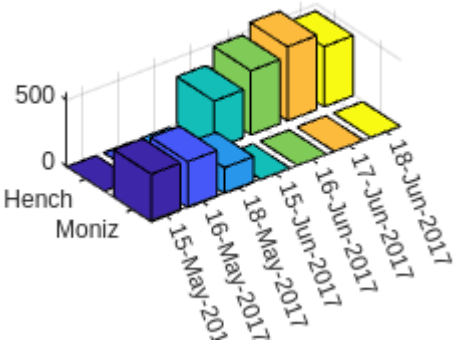


Replace the hard-coded values that set the y tick labels

```
mouse_names =unique(data_summer2017.mouse);
session_dates = unique(data_summer2017.session_date);

mouse_session_counts=groupcounts({data_summer2017.mouse,
data_summer2017.session_date}, "IncludeEmptyGroups",true);


reshaped_data_for_plot =
reshape(mouse_session_counts,length(session_dates),length(mouse_names))';
bar3(reshaped_data_for_plot)
yticklabels(mouse_names)
xticks([1:1: 7])
xticklabels(datestr(session_dates))
```



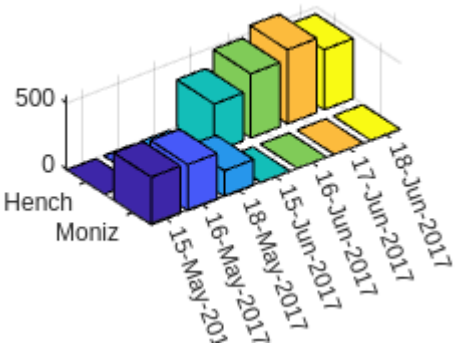
Lastly, remove the hard-coding that sets the positions of the xticks. **Hint** - replace the value 7 with a `length()` function

```
mouse_names =unique(data_summer2017.mouse);
```

```
session_dates = unique(data_summer2017.session_date);

mouse_session_counts=groupcounts({data_summer2017.mouse,
data_summer2017.session_date}, "IncludeEmptyGroups",true);


reshaped_data_for_plot =
reshape(mouse_session_counts,length(session_dates),length(mouse_names))';
bar3(reshaped_data_for_plot)
yticklabels(mouse_names)
xticks([1:1: length(session_dates)])
xticklabels(datestr(session_dates))
```



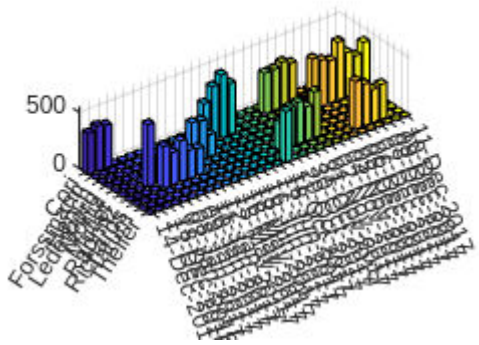
Now you have built a recipe that can make a similar plot for any of the datasets! Make the plot for the complete dataset, `data`

```
data = [data_winter2016; data_winter2017; data_summer2017];
```

```
mouse_names =unique(data.mouse);
session_dates = unique(data.session_date);

mouse_session_counts=groupcounts({data.mouse, data.session_date},
"IncludeEmptyGroups",true);

reshaped_data_for_plot =
reshape(mouse_session_counts,length(session_dates),length(mouse_names))';
bar3(reshaped_data_for_plot)
yticklabels(mouse_names)
xticks([1:1: length(session_dates)])
xticklabels(datestr(session_dates))
```



Add an appropriate label for the z axis to your plot

```
mouse_names =unique(data.mouse);
session_dates = unique(data.session_date);

mouse_session_counts=groupcounts({data.mouse, data.session_date},
"IncludeEmptyGroups",true);

reshaped_data_for_plot =
reshape(mouse_session_counts,length(session_dates),length(mouse_names))';
bar3(reshaped_data_for_plot)
yticklabels(mouse_names)
xticks([1:1: length(session_dates)])
xticklabels(datestr(session_dates))
zlabel('num. trials')
```