Table of Contents


```
clear all; clc; close all
```

Task 1 Data

```
% load data
load('/home/tem/Documents/MATLAB/BMI/KD/6-28-19_Tem/MAT_Data/
jaco_task1_translation.mat');
load('/home/tem/Documents/MATLAB/BMI/KD/6-28-19_Tem/MAT_Data/
jaco_task1_rotation.mat');

% Convert Quaternions to Euler Angles
jaco_task1_rotation = quat2eul(jaco_task1_rotation, 'XYZ');

% Separate data by trial
msg = 1;
ind = [770; 972; 1065; 1200; 1099]; % number of datapoints per trial
 (/tf messages)
jaco_task1_trials = {};

for i = 1:length(ind)
    jaco_task1_trials{end+1} =
 [jaco_task1_translation(msg:sum(ind(1:i)),:)
 jaco_task1_rotation(msg:sum(ind(1:i)),:)];
    msg = msg + ind(i);
end

% Find Changes in Position and Orientation
jaco_task1_vel = {[] [] [] [] []};

for i = 1:length(ind)
    for j = 1:(size(jaco_task1_trials{i},1)-1)
```

```
        vel = jaco_task1_trials{i}(j+1,:) - jaco_task1_trials{i}
(j,:); % position and orientation change

        if max(abs(vel(1:3))) > 3e-3 || max(abs(vel(4:6))) > 2e-2    %
 ignore small commands
            jaco_task1_vel{i} = [jaco_task1_vel{i}; vel];
        end
    end
end

% Proportion of Axes Used for Each Trial
count1 = zeros(5,6); % count commands from axes [x y z X Y Z] for each
 trial
percent1 = zeros(5,6); %  proportion of points axis used : trial(row)
 vs axis(column)

pts = [144; 238; 256; 216; 197]; % number of data points for each
 trial

for i = 1:length(ind)

    for j = 1:size(jaco_task1_vel{i},1)
        for k = 1:size(jaco_task1_vel{i},2)
            if k >= 4
                if abs(jaco_task1_vel{i}(j,k)) > 2e-2   % ignore small
 commands for rotation axes
                    count1(i,k) = count1(i,k) + 1;
                end
            else
                if abs(jaco_task1_vel{i}(j,k)) > 3e-3   % ignore small
 commands for translation axes
                    count1(i,k) = count1(i,k) + 1;
                end
            end
        end
    end
    percent1(i,:) = count1(i,:) ./ pts(i);

end
```

Task 2 Data

```
% load data
load('/home/tem/Documents/MATLAB/BMI/KD/6-28-19_Tem/MAT_Data/
jaco_task2_translation.mat');
load('/home/tem/Documents/MATLAB/BMI/KD/6-28-19_Tem/MAT_Data/
jaco_task2_rotation.mat');

% Convert Quaternions to Euler Angles
jaco_task2_rotation = quat2eul(jaco_task2_rotation, 'XYZ');

% Separate data by trial
msg = 1;
```

```
ind = [835; 847; 758; 774; 897]; % number of datapoints per trial (/tf
 messages)
jaco_task2_trials = {};

for i = 1:length(ind)
    jaco_task2_trials{end+1} =
 [jaco_task2_translation(msg:sum(ind(1:i)),:)
 jaco_task2_rotation(msg:sum(ind(1:i)),:)];
    msg = msg + ind(i);
end

% Find Changes in Position and Orientation
jaco_task2_vel = {[] [] [] [] []};

for i = 1:length(ind)
    for j = 1:(size(jaco_task2_trials{i},1)-1)

        vel = jaco_task2_trials{i}(j+1,:) - jaco_task2_trials{i}
(j,:); % position and orientation change

        if max(abs(vel(1:3))) > 3e-3 || max(abs(vel(4:6))) > 2e-2
  % ignore small commands
            jaco_task2_vel{i} = [jaco_task2_vel{i}; vel];
        end
    end
end

% Proportion of Axes Used for Each Trial
count2 = zeros(5,6); % count commands from axes [x y z X Y Z] for each
 trial
percent2 = zeros(5,6); %  proportion of points axis used : trial(row)
 vs axis(column)

pts = [92; 93; 114; 97; 97]; % number of data points for each trial

for i = 1:length(ind)

    for j = 1:size(jaco_task2_vel{i},1)
        for k = 1:size(jaco_task2_vel{i},2)
            if k >= 4
                if abs(jaco_task2_vel{i}(j,k)) > 2e-2   % ignore small
 commands for rotation axes
                    count2(i,k) = count2(i,k) + 1;
                end
            else
                if abs(jaco_task2_vel{i}(j,k)) > 3e-3   % ignore small
 commands for translation axes
                    count2(i,k) = count2(i,k) + 1;
                end
            end
        end
    end
    percent2(i,:) = count2(i,:) ./ pts(i);
```

```
end
```

Task 3 Data

```
% load data
load('/home/tem/Documents/MATLAB/BMI/KD/6-28-19_Tem/MAT_Data/
jaco_task3_translation.mat');
load('/home/tem/Documents/MATLAB/BMI/KD/6-28-19_Tem/MAT_Data/
jaco_task3_rotation.mat');

% Convert Quaternions to Euler Angles
jaco_task3_rotation = quat2eul(jaco_task3_rotation, 'XYZ');

% Separate data by trial
msg = 1;
ind = [688; 696; 643; 559; 553]; % number of datapoints per trial (/tf
 messages)
jaco_task3_trials = {};

for i = 1:length(ind)
    jaco_task3_trials{end+1} =
 [jaco_task3_translation(msg:sum(ind(1:i)),:)
 jaco_task3_rotation(msg:sum(ind(1:i)),:)];
    msg = msg + ind(i);
end

% Find Changes in Position and Orientation
jaco_task3_vel = {[] [] [] [] []};

for i = 1:length(ind)
    for j = 1:(size(jaco_task3_trials{i},1)-1)

        vel = jaco_task3_trials{i}(j+1,:) - jaco_task3_trials{i}
(j,:); % position and orientation change

        if max(abs(vel(1:3))) > 3e-3 || max(abs(vel(4:6))) > 2e-2
  % ignore small commands
            jaco_task3_vel{i} = [jaco_task3_vel{i}; vel];
        end
    end
end

% Proportion of Axes Used for Each Trial
count3 = zeros(5,6); % count commands from axes [x y z X Y Z] for each
 trial
percent3 = zeros(5,6); %  proportion of points axis used : trial(row)
 vs axis(column)

pts = [67; 88; 76; 98; 107]; % number of data points for each trial

for i = 1:length(ind)

    for j = 1:size(jaco_task3_vel{i},1)
```

```
        for k = 1:size(jaco_task3_vel{i},2)
            if k >= 4
                if abs(jaco_task3_vel{i}(j,k)) > 2e-2   % ignore small
 commands for rotation axes
                    count3(i,k) = count3(i,k) + 1;
                end
            else
                if abs(jaco_task3_vel{i}(j,k)) > 3e-3   % ignore small
 commands for translation axes
                    count3(i,k) = count3(i,k) + 1;
                end
            end
        end
    end
    percent3(i,:) = count3(i,:) ./ pts(i);

end
```

Task 4 Data

```
% load data
load('/home/tem/Documents/MATLAB/BMI/KD/6-28-19_Tem/MAT_Data/
jaco_task4_translation.mat');
load('/home/tem/Documents/MATLAB/BMI/KD/6-28-19_Tem/MAT_Data/
jaco_task4_rotation.mat');

% Convert Quaternions to Euler Angles
jaco_task4_rotation = quat2eul(jaco_task4_rotation, 'XYZ');

% Separate data by trial
msg = 1;
ind = [871; 858; 915; 894; 1016]; % number of datapoints per trial (/
tf messages)
jaco_task4_trials = {};

for i = 1:length(ind)
    jaco_task4_trials{end+1} =
 [jaco_task4_translation(msg:sum(ind(1:i)),:)
 jaco_task4_rotation(msg:sum(ind(1:i)),:)];
    msg = msg + ind(i);
end

% Find Changes in Position and Orientation
jaco_task4_vel = {[] [] [] [] []};

for i = 1:length(ind)
    for j = 1:(size(jaco_task4_trials{i},1)-1)

        vel = jaco_task4_trials{i}(j+1,:) - jaco_task4_trials{i}
(j,:); % position change

        if max(abs(vel(1:3))) > 3e-3 || max(abs(vel(4:6))) > 2e-2
  % ignore small commands
```

```
            jaco_task4_vel{i} = [jaco_task4_vel{i}; vel]; %
 translation
        end
    end
end

% Proportion of Axes Used for Each Trial
count4 = zeros(5,6); % count commands from axes [x y z X Y Z] for each
 trial
percent4 = zeros(5,6); %  proportion of points axis used : trial(row)
 vs axis(column)

pts = [142; 114; 173; 71; 133]; % number of data points for each trial

for i = 1:length(ind)

    for j = 1:size(jaco_task4_vel{i},1)
        for k = 1:size(jaco_task4_vel{i},2)
            if k >= 4
                if abs(jaco_task4_vel{i}(j,k)) > 2e-2   % ignore small
 commands for rotation axes
                    count4(i,k) = count4(i,k) + 1;
                end
            else
                if abs(jaco_task4_vel{i}(j,k)) > 3e-3   % ignore small
 commands for translation axes
                    count4(i,k) = count4(i,k) + 1;
                end
            end
        end
    end
    percent4(i,:) = count4(i,:) ./ pts(i);

end
```

Task 5 Data

```
% load data
load('/home/tem/Documents/MATLAB/BMI/KD/6-28-19_Tem/MAT_Data/
jaco_task5_translation.mat');
load('/home/tem/Documents/MATLAB/BMI/KD/6-28-19_Tem/MAT_Data/
jaco_task5_rotation.mat');

% Convert Quaternions to Euler Angles
jaco_task5_rotation = quat2eul(jaco_task5_rotation, 'XYZ');

% Separate data by trial
msg = 1;
ind = [1048; 1090; 975; 1019; 907]; % number of datapoints per trial
 (/tf messages)
jaco_task5_trials = {};

for i = 1:length(ind)
```

```
    jaco_task5_trials{end+1} =
 [jaco_task5_translation(msg:sum(ind(1:i)),:)
 jaco_task5_rotation(msg:sum(ind(1:i)),:)];
    msg = msg + ind(i);
end

% Find Changes in Position and Orientation
jaco_task5_vel = {[] [] [] [] []};

for i = 1:length(ind)
    for j = 1:(size(jaco_task5_trials{i},1)-1)

        vel = jaco_task5_trials{i}(j+1,:) - jaco_task5_trials{i}
(j,:); % position change

        if max(abs(vel(1:3))) > 3e-3 || max(abs(vel(4:6))) > 2e-2
  % ignore small commands
            jaco_task5_vel{i} = [jaco_task5_vel{i}; vel]; %
 translation
        end
    end
end

% Proportion of Axes Used for Each Trial
count5 = zeros(5,6); % count commands from axes [x y z X Y Z] for each
 trial
percent5 = zeros(5,6); %  proportion of points axis used : trial(row)
 vs axis(column)

pts = [153; 212; 182; 243; 205]; % number of data points for each
 trial

for i = 1:length(ind)

    for j = 1:size(jaco_task5_vel{i},1)
        for k = 1:size(jaco_task5_vel{i},2)
            if k >= 4
                if abs(jaco_task5_vel{i}(j,k)) > 2e-2   % ignore small
 commands for translation axes
                    count5(i,k) = count5(i,k) + 1;
                end
            else
                if abs(jaco_task5_vel{i}(j,k)) > 3e-3   % ignore small
 commands for translation axes
                    count5(i,k) = count5(i,k) + 1;
                end
            end
        end
    end
    percent5(i,:) = count5(i,:) ./ pts(i);

end
```

Calculate Means, Standard Deviations and Remove Artifacts

```
% Calculate Means and Standard Deviations for Activation Percentage
percent = {percent1 percent2 percent3 percent4 percent5};

means = [];
stds = [];
for i = 1:5
    means = [means; mean(percent{i},1)];
    stds = [stds; std(percent{i},0,1)];
end

% Reorganize data into matrices for each task
jaco_task1_vel = [jaco_task1_vel{1}; jaco_task1_vel{2};
 jaco_task1_vel{3}; jaco_task1_vel{4}; jaco_task1_vel{5}];
jaco_task2_vel = [jaco_task2_vel{1}; jaco_task2_vel{2};
 jaco_task2_vel{3}; jaco_task2_vel{4}; jaco_task2_vel{5}];
jaco_task3_vel = [jaco_task3_vel{1}; jaco_task3_vel{2};
 jaco_task3_vel{3}; jaco_task3_vel{4}; jaco_task3_vel{5}];
jaco_task4_vel = [jaco_task4_vel{1}; jaco_task4_vel{2};
 jaco_task4_vel{3}; jaco_task4_vel{4}; jaco_task4_vel{5}];
jaco_task5_vel = [jaco_task5_vel{1}; jaco_task5_vel{2};
 jaco_task5_vel{3}; jaco_task5_vel{4}; jaco_task5_vel{5}];

% Get rid of any values that are too small for boxplots
for i = 1:6
    for j = 1:size(jaco_task1_vel,1)
        if i <= 3
            if abs(jaco_task1_vel(j,i)) < 3e-3  % ignore small
 translations
                jaco_task1_vel(j,i) = NaN;
            end
        else
            if abs(jaco_task1_vel(j,i)) < 2e-2 % ignore small
 rotations
                jaco_task1_vel(j,i) = NaN;
            end
        end
    end

    for j = 1:size(jaco_task2_vel,1)
        if i <= 3
            if abs(jaco_task2_vel(j,i)) < 3e-3  % ignore small
 translations
                jaco_task2_vel(j,i) = NaN;
            end
        else
            if abs(jaco_task2_vel(j,i)) < 2e-2 % ignore small
 rotations
                jaco_task2_vel(j,i) = NaN;
            end
```

```
        end
    end

    for j = 1:size(jaco_task3_vel,1)
        if i <= 3
            if abs(jaco_task3_vel(j,i)) < 3e-3  % ignore small
 translations
                jaco_task3_vel(j,i) = NaN;
            end
        else
            if abs(jaco_task3_vel(j,i)) < 2e-2 % ignore small
 rotations
                jaco_task3_vel(j,i) = NaN;
            end
        end
    end

    for j = 1:size(jaco_task4_vel,1)
        if i <= 3
            if abs(jaco_task4_vel(j,i)) < 3e-3  % ignore small
 translations
                jaco_task4_vel(j,i) = NaN;
            end
        else
            if abs(jaco_task4_vel(j,i)) < 2e-2 % ignore small
 rotations
                jaco_task4_vel(j,i) = NaN;
            end
        end
    end

    for j = 1:size(jaco_task5_vel,1)
        if i <= 3
            if abs(jaco_task5_vel(j,i)) < 3e-3  % ignore small
 translations
                jaco_task5_vel(j,i) = NaN;
            end
        else
            if abs(jaco_task5_vel(j,i)) < 2e-2 % ignore small
 rotations
                jaco_task5_vel(j,i) = NaN;
            end
        end
    end
end

% Get rid of any artifacts for boxplots
for i = 1:size(jaco_task1_vel,1)
    for j = 1:size(jaco_task1_vel,2)
        if abs(jaco_task1_vel(i,j)) > 1
            jaco_task1_vel(i,j) = NaN;
        end
    end
end
```

```
for i = 1:size(jaco_task2_vel,1)
    for j = 1:size(jaco_task2_vel,2)
        if abs(jaco_task2_vel(i,j)) > 1
            jaco_task2_vel(i,j) = NaN;
        end
    end
end

for i = 1:size(jaco_task3_vel,1)
    for j = 1:size(jaco_task3_vel,2)
        if abs(jaco_task3_vel(i,j)) > 1
            jaco_task3_vel(i,j) = NaN;
        end
    end
end

for i = 1:size(jaco_task4_vel,1)
    for j = 1:size(jaco_task4_vel,2)
        if abs(jaco_task4_vel(i,j)) > 1
            jaco_task4_vel(i,j) = NaN;
        end
    end
end

for i = 1:size(jaco_task5_vel,1)
    for j = 1:size(jaco_task5_vel,2)
        if abs(jaco_task5_vel(i,j)) > 1
            jaco_task5_vel(i,j) = NaN;
        end
    end
end
```

Calculate P-Values and Distributions

```
% Make task matrices the same dimensions
jaco_task2_vel = [jaco_task2_vel; nan(558,6)];
jaco_task3_vel = [jaco_task3_vel; nan(615,6)];
jaco_task4_vel = [jaco_task4_vel; nan(418,6)];
jaco_task5_vel = [jaco_task5_vel; nan(56,6)];

% Reorganize data into matrices for each axis
jaco_trans_x = [jaco_task1_vel(:,1) jaco_task2_vel(:,1)
 jaco_task3_vel(:,1) jaco_task4_vel(:,1) jaco_task5_vel(:,1)];
jaco_trans_y = [jaco_task1_vel(:,2) jaco_task2_vel(:,2)
 jaco_task3_vel(:,2) jaco_task4_vel(:,2) jaco_task5_vel(:,2)];
jaco_trans_z = [jaco_task1_vel(:,3) jaco_task2_vel(:,3)
 jaco_task3_vel(:,3) jaco_task4_vel(:,3) jaco_task5_vel(:,3)];
jaco_rot_x = [jaco_task1_vel(:,4) jaco_task2_vel(:,4)
 jaco_task3_vel(:,4) jaco_task4_vel(:,4) jaco_task5_vel(:,4)];
jaco_rot_y = [jaco_task1_vel(:,5) jaco_task2_vel(:,5)
 jaco_task3_vel(:,5) jaco_task4_vel(:,5) jaco_task5_vel(:,5)];
```

```
jaco_rot_z = [jaco_task1_vel(:,6) jaco_task2_vel(:,6)
 jaco_task3_vel(:,6) jaco_task4_vel(:,6) jaco_task5_vel(:,6)];

% Trials (Rows) vs Tasks (Columns) for each axis
percent_trans_x = [percent1(:,1) percent2(:,1) percent3(:,1)
 percent4(:,1) percent5(:,1)];
percent_trans_y = [percent1(:,2) percent2(:,2) percent3(:,2)
 percent4(:,2) percent5(:,2)];
percent_trans_z = [percent1(:,3) percent2(:,3) percent3(:,3)
 percent4(:,3) percent5(:,3)];
percent_rot_x = [percent1(:,4) percent2(:,4) percent3(:,4)
 percent4(:,4) percent5(:,4)];
percent_rot_y = [percent1(:,5) percent2(:,5) percent3(:,5)
 percent4(:,5) percent5(:,5)];
percent_rot_z = [percent1(:,6) percent2(:,6) percent3(:,6)
 percent4(:,6) percent5(:,6)];

% Variables needed for significance comparison between tasks
task_group = {[1,2],[1,3],[1,4],[1,5],[2,3],[2,4],[2,5],[3,4],[3,5],
[4,5]};
tx = {}; ty = {}; tz = {}; rx = {}; ry = {}; rz = {};
atx = {}; aty = {}; atz = {}; arx = {}; ary = {}; arz = {};

% ANOVA Test for 2 tasks on the same axis
p_tx = []; p_ty = []; p_tz = []; % Command Value
p_rx = []; p_ry = []; p_rz = [];

p_atx = []; p_aty = []; p_atz = []; % Activation Percentage
p_arx = []; p_ary = []; p_arz = [];

for i = 1:length(task_group)
    g = task_group{i};

    j_tx = [jaco_trans_x(:,g(1)) jaco_trans_x(:,g(2))];
    j_ty = [jaco_trans_y(:,g(1)) jaco_trans_y(:,g(2))];
    j_tz = [jaco_trans_z(:,g(1)) jaco_trans_z(:,g(2))];
    j_rx = [jaco_rot_x(:,g(1)) jaco_rot_x(:,g(2))];
    j_ry = [jaco_rot_y(:,g(1)) jaco_rot_y(:,g(2))];
    j_rz = [jaco_rot_z(:,g(1)) jaco_rot_z(:,g(2))];

    % Ignore groups with p-values higher than 0.05
    if anova1(j_tx,g,'off') <= 0.05
        p_tx = [p_tx; anova1(j_tx,g,'off')];
        tx{end+1} = g;
    end

    if anova1(j_ty,g,'off') <= 0.05
        p_ty = [p_ty; anova1(j_ty,g,'off')];
        ty{end+1} = g;
    end

    if anova1(j_tz,g,'off') <= 0.05
        p_tz = [p_tz; anova1(j_tz,g,'off')];
        tz{end+1} = g;
```

```
end

if anova1(j_rx,g,'off') <= 0.05
    p_rx = [p_rx; anova1(j_rx,g,'off')];
    rx{end+1} = g;
end

if anova1(j_ry,g,'off') <= 0.05
    p_ry = [p_ry; anova1(j_ry,g,'off')];
    ry{end+1} = g;
end

if anova1(j_rz,g,'off') <= 0.05
    p_rz = [p_rz; anova1(j_rz,g,'off')];
    rz{end+1} = g;
end

pc_tx = [percent_trans_x(:,g(1)) percent_trans_x(:,g(2))];
pc_ty = [percent_trans_y(:,g(1)) percent_trans_y(:,g(2))];
pc_tz = [percent_trans_z(:,g(1)) percent_trans_z(:,g(2))];
pc_rx = [percent_rot_x(:,g(1)) percent_rot_x(:,g(2))];
pc_ry = [percent_rot_y(:,g(1)) percent_rot_y(:,g(2))];
pc_rz = [percent_rot_z(:,g(1)) percent_rot_z(:,g(2))];

if anova1(pc_tx,g,'off') <= 0.05
    p_atx = [p_atx; anova1(pc_tx,g,'off')];
    atx{end+1} = g;
end

if anova1(pc_ty,g,'off') <= 0.05
    p_aty = [p_aty; anova1(pc_ty,g,'off')];
    aty{end+1} = g;
end

if anova1(pc_tz,g,'off') <= 0.05
    p_atz = [p_atz; anova1(pc_tz,g,'off')];
    atz{end+1} = g;
end

if anova1(pc_rx,g,'off') <= 0.05
    p_arx = [p_arx; anova1(pc_rx,g,'off')];
    arx{end+1} = g;
end

if anova1(pc_ry,g,'off') <= 0.05
    p_ary = [p_ary; anova1(pc_ry,g,'off')];
    ary{end+1} = g;
end

if anova1(pc_rz,g,'off') <= 0.05
    p_arz = [p_arz; anova1(pc_rz,g,'off')];
    arz{end+1} = g;
end
```

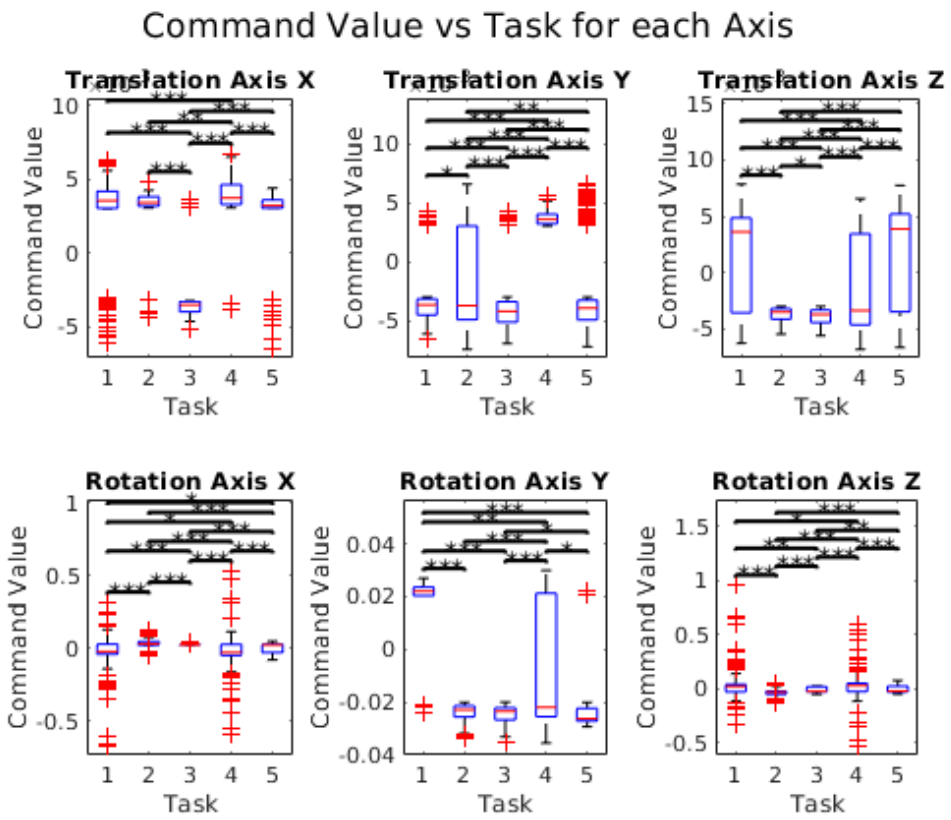
```
end

% Variables needed for significance comparison between axes
axes_group = {[1,2],[1,3],[1,4],[1,5],[1,6],[2,3],[2,4],[2,5],[2,6],
[3,4],[3,5],[3,6],[4,5],[4,6],[5,6]};
t1 = {}; t2 = {}; t3 = {}; t4 = {}; t5 = {};
at1 = {}; at2 = {}; at3 = {}; at4 = {}; at5 = {};

% ANOVA Test for 2 axes in the same task
p_t1 = []; p_t2 = []; p_t3 = []; p_t4 = []; p_t5 = []; % Command Value
p_at1 = []; p_at2 = []; p_at3 = []; p_at4 = []; p_at5 = []; %
 Activation Percentage

for i = 1:length(axes_group)
    g = axes_group{i};

    j_t1 = [jaco_task1_vel(:,g(1)) jaco_task1_vel(:,g(2))];
    j_t2 = [jaco_task2_vel(:,g(1)) jaco_task2_vel(:,g(2))];
    j_t3 = [jaco_task3_vel(:,g(1)) jaco_task3_vel(:,g(2))];
    j_t4 = [jaco_task4_vel(:,g(1)) jaco_task4_vel(:,g(2))];
    j_t5 = [jaco_task5_vel(:,g(1)) jaco_task5_vel(:,g(2))];

    % Ignore groups with p-values higher than 0.05
    if anova1(j_t1,g,'off') <= 0.05
        p_t1 = [p_t1; anova1(j_t1,g,'off')];
        t1{end+1} = g;
    end

    if anova1(j_t2,g,'off') <= 0.05
        p_t2 = [p_t2; anova1(j_t2,g,'off')];
        t2{end+1} = g;
    end

    if anova1(j_t3,g,'off') <= 0.05
        p_t3 = [p_t3; anova1(j_t3,g,'off')];
        t3{end+1} = g;
    end

    if anova1(j_t4,g,'off') <= 0.05
        p_t4 = [p_t4; anova1(j_t4,g,'off')];
        t4{end+1} = g;
    end

    if anova1(j_t5,g,'off') <= 0.05
        p_t5 = [p_t5; anova1(j_t5,g,'off')];
        t5{end+1} = g;
    end

    pc_t1 = [percent1(:,g(1)) percent1(:,g(2))];
    pc_t2 = [percent2(:,g(1)) percent2(:,g(2))];
    pc_t3 = [percent3(:,g(1)) percent3(:,g(2))];
    pc_t4 = [percent4(:,g(1)) percent4(:,g(2))];
    pc_t5 = [percent5(:,g(1)) percent5(:,g(2))];
```

```
        % Ignore groups with p-values higher than 0.05
        if anova1(pc_t1,g,'off') <= 0.05
            p_at1 = [p_at1; anova1(pc_t1,g,'off')];
            at1{end+1} = g;
        end

        if anova1(pc_t2,g,'off') <= 0.05
            p_at2 = [p_at2; anova1(pc_t2,g,'off')];
            at2{end+1} = g;
        end

        if anova1(pc_t3,g,'off') <= 0.05
            p_at3 = [p_at3; anova1(pc_t3,g,'off')];
            at3{end+1} = g;
        end

        if anova1(pc_t4,g,'off') <= 0.05
            p_at4 = [p_at4; anova1(pc_t4,g,'off')];
            at4{end+1} = g;
        end

        if anova1(pc_t5,g,'off') <= 0.05
            p_at5 = [p_at5; anova1(pc_t5,g,'off')];
            at5{end+1} = g;
        end

    end
```

Command Value vs Task for each Axis

```
    % Plot for each Axis
    sgtitle('Command Value vs Task for each Axis')

    subplot(2,3,1)
    boxplot(jaco_trans_x)
    xlim([0.5 5.5])
    xticks([1 2 3 4 5])
    xticklabels({'1','2','3','4','5'})
    xlabel('Task')
    ylabel('Command Value')
    title('Translation Axis X')
    hold on
    sigstar(tx, p_tx)
    hold off

    subplot(2,3,2)
    boxplot(jaco_trans_y)
    xlim([0.5 5.5])
    xticks([1 2 3 4 5])
    xticklabels({'1','2','3','4','5'})
    xlabel('Task')
    ylabel('Command Value')
    title('Translation Axis Y')
```

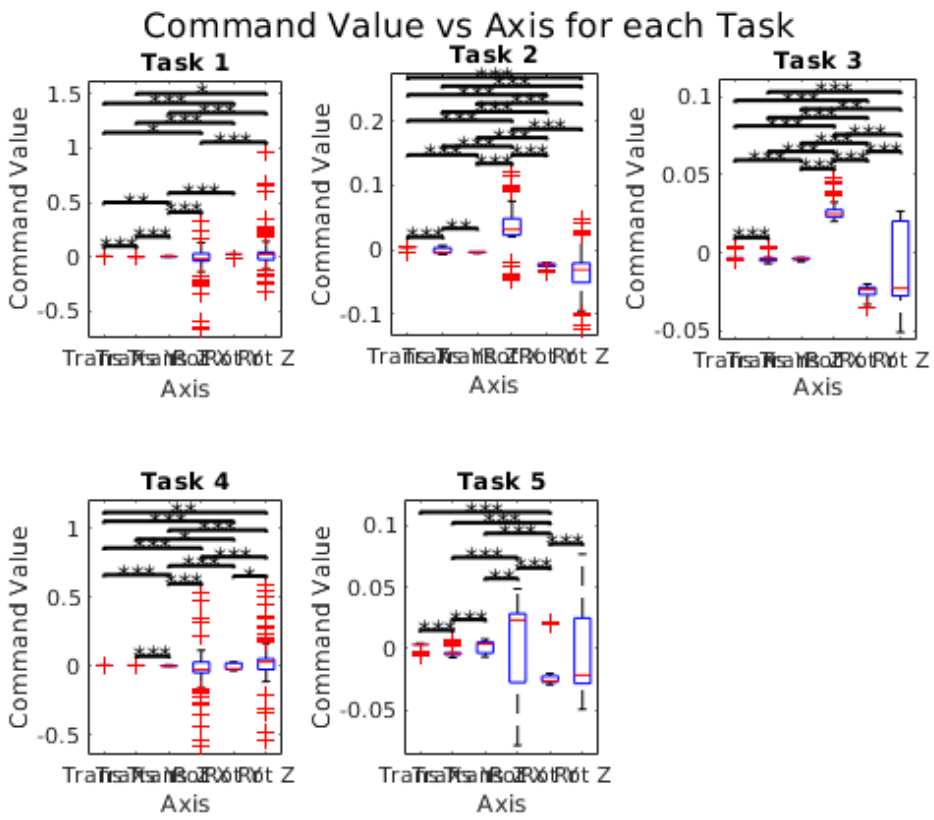
```
hold on
sigstar(ty, p_ty)
hold off

subplot(2,3,3)
boxplot(jaco_trans_z)
xlim([0.5 5.5])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Command Value')
title('Translation Axis Z')
hold on
sigstar(tz, p_tz)
hold off

subplot(2,3,4)
boxplot(jaco_rot_x)
xlim([0.5 5.5])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Command Value')
title('Rotation Axis X')
hold on
sigstar(rx, p_rx)
hold off

subplot(2,3,5)
boxplot(jaco_rot_y)
xlim([0.5 5.5])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Command Value')
title('Rotation Axis Y')
hold on
sigstar(ry, p_ry)
hold off

subplot(2,3,6)
boxplot(jaco_rot_z)
xlim([0.5 5.5])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Command Value')
title('Rotation Axis Z')
hold on
sigstar(rz, p_rz)
hold off
```



Command Value vs Axis for each Task

```
% Plot for each Task
figure(2)

sgtitle('Command Value vs Axis for each Task')

subplot(2,3,1)
boxplot(jaco_task1_vel)
xticklabels({'Trans X','Trans Y','Trans Z','Rot X','Rot Y','Rot Z'})
xlabel('Axis')
ylabel('Command Value')
title('Task 1')
hold on
sigstar(t1, p_t1)
hold off

subplot(2,3,2)
boxplot(jaco_task2_vel)
xticklabels({'Trans X','Trans Y','Trans Z','Rot X','Rot Y','Rot Z'})
xlabel('Axis')
ylabel('Command Value')
title('Task 2')
hold on
```

```
sigstar(t2, p_t2)
hold off

subplot(2,3,3)
boxplot(jaco_task3_vel)
xticklabels({'Trans X','Trans Y','Trans Z','Rot X','Rot Y','Rot Z'})
xlabel('Axis')
ylabel('Command Value')
title('Task 3')
hold on
sigstar(t3, p_t3)
hold off

subplot(2,3,4)
boxplot(jaco_task4_vel)
xticklabels({'Trans X','Trans Y','Trans Z','Rot X','Rot Y','Rot Z'})
xlabel('Axis')
ylabel('Command Value')
title('Task 4')
hold on
sigstar(t4, p_t4)
hold off

subplot(2,3,5)
boxplot(jaco_task5_vel)
xticklabels({'Trans X','Trans Y','Trans Z','Rot X','Rot Y','Rot Z'})
xlabel('Axis')
ylabel('Command Value')
title('Task 5')
hold on
sigstar(t5, p_t5)
hold off
```

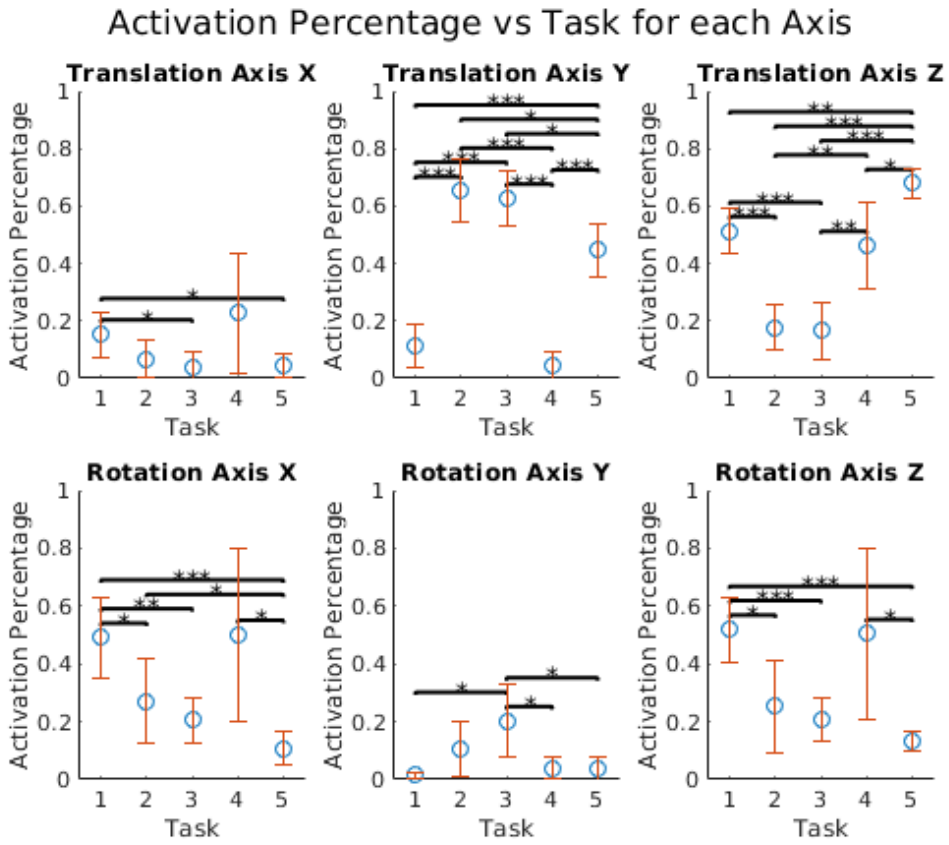


Activation Percentage vs Task for Each Axis

```
figure(3)

subplot(2,3,1)
scatter(1:5, means(:,1))
axis([0.5 5.5 0 1])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Activation Percentage')
title('Translation Axis X')
hold on
sigstar(atx, p_atx)
errorbar(1:5, means(:,1), stds(:,1), 'LineStyle', 'None')
hold off

subplot(2,3,2)
scatter(1:5, means(:,2))
axis([0.5 5.5 0 1])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Activation Percentage')
```

```
title('Translation Axis Y')
hold on
sigstar(aty, p_aty)
errorbar(1:5, means(:,2), stds(:,2), 'LineStyle', 'None')
hold off

subplot(2,3,3)
scatter(1:5, means(:,3))
axis([0.5 5.5 0 1])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Activation Percentage')
title('Translation Axis Z')
hold on
sigstar(atz, p_atz)
errorbar(1:5, means(:,3), stds(:,3), 'LineStyle', 'None')
hold off

subplot(2,3,4)
scatter(1:5, means(:,4))
axis([0.5 5.5 0 1])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Activation Percentage')
title('Rotation Axis X')
hold on
sigstar(arx, p_arx)
errorbar(1:5, means(:,4), stds(:,4), 'LineStyle', 'None')
hold off

subplot(2,3,5)
scatter(1:5, means(:,5))
axis([0.5 5.5 0 1])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Activation Percentage')
title('Rotation Axis Y')
hold on
sigstar(ary, p_ary)
errorbar(1:5, means(:,5), stds(:,5), 'LineStyle', 'None')
hold off

subplot(2,3,6)
scatter(1:5, means(:,6))
axis([0.5 5.5 0 1])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Activation Percentage')
title('Rotation Axis Z')
hold on
```

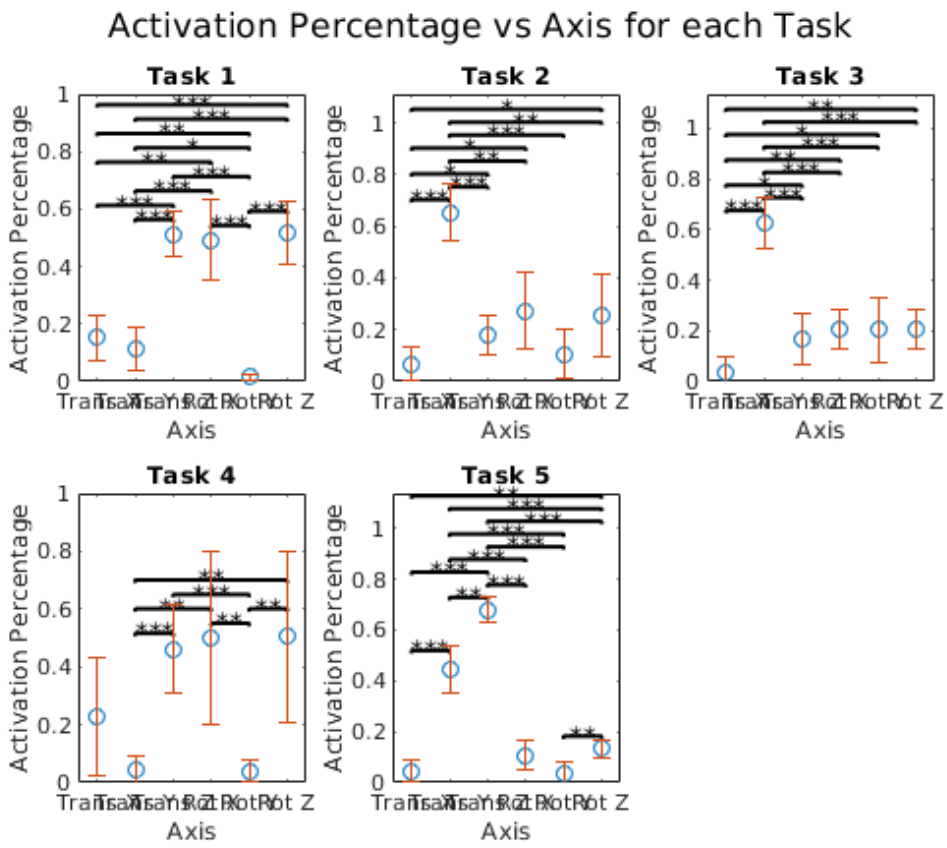
```
sigstar(arz, p_arz)
errorbar(1:5, means(:,6), stds(:,6), 'LineStyle', 'None')
hold off

sgtitle('Activation Percentage vs Task for each Axis')
```



Activation Percentage vs Axis for each Task

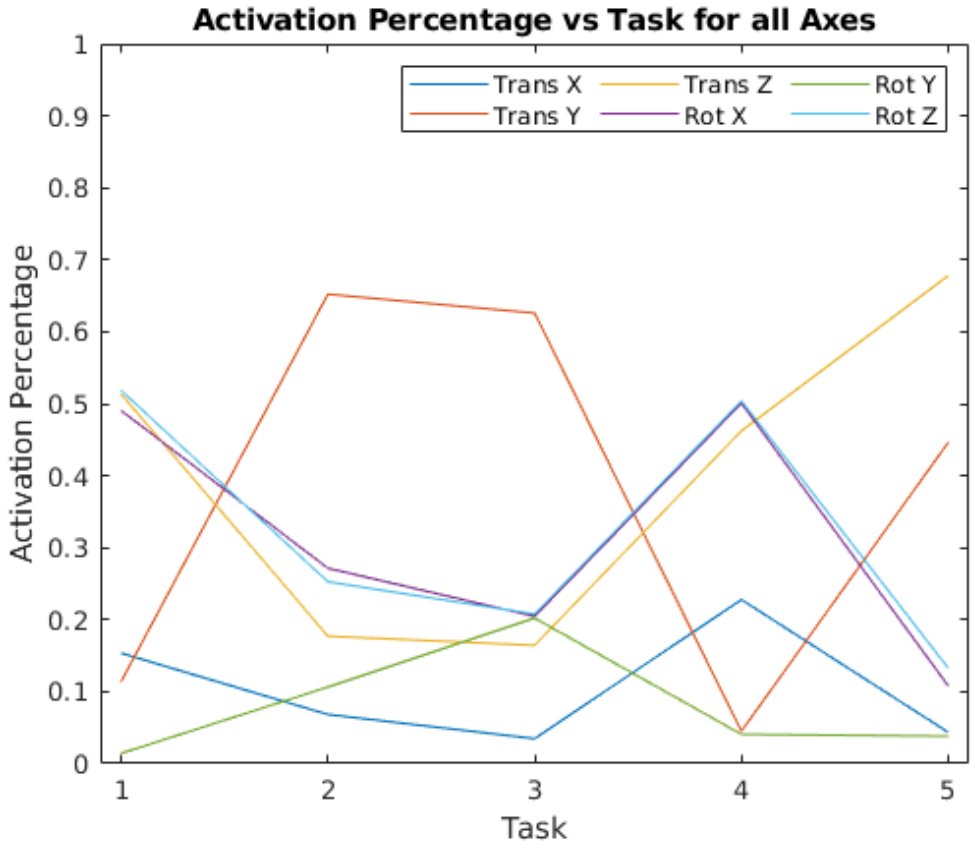
```
figure(4)

subplot (2,3,1)
plot(means(1,:),'LineStyle', 'None','Marker','o')
axis([0.5 6.5 0 1])
xticks([1 2 3 4 5 6])
xticklabels({'Trans X','Trans Y','Trans Z','Rot X','Rot Y','Rot Z'})
xlabel('Axis')
ylabel('Activation Percentage')
title('Task 1')
hold on
sigstar(at1, p_at1)
errorbar(means(1,:), stds(1,:),'LineStyle','None')
hold off

subplot (2,3,2)
```

```
plot(means(2,:),'LineStyle','None','Marker','o')
axis([0.5 6.5 0 1])
xticks([1 2 3 4 5 6])
xticklabels({'Trans X','Trans Y','Trans Z','Rot X','Rot Y','Rot Z'})
xlabel('Axis')
ylabel('Activation Percentage')
title('Task 2')
hold on
sigstar(at2, p_at2)
errorbar(means(2,:), stds(2,:),'LineStyle','None')
hold off

subplot (2,3,3)
plot(means(3,:),'LineStyle','None','Marker','o')
axis([0.5 6.5 0 1])
xticks([1 2 3 4 5 6])
xticklabels({'Trans X','Trans Y','Trans Z','Rot X','Rot Y','Rot Z'})
xlabel('Axis')
ylabel('Activation Percentage')
title('Task 3')
hold on
sigstar(at3, p_at3)
errorbar(means(3,:), stds(3,:),'LineStyle','None')
hold off

subplot (2,3,4)
plot(means(4,:),'LineStyle','None','Marker','o')
axis([0.5 6.5 0 1])
xticks([1 2 3 4 5 6])
xticklabels({'Trans X','Trans Y','Trans Z','Rot X','Rot Y','Rot Z'})
xlabel('Axis')
ylabel('Activation Percentage')
title('Task 4')
hold on
sigstar(at4, p_at4)
errorbar(means(4,:), stds(4,:),'LineStyle','None')
hold off

subplot (2,3,5)
plot(means(5,:),'LineStyle','None','Marker','o')
axis([0.5 6.5 0 1])
xticks([1 2 3 4 5 6])
xticklabels({'Trans X','Trans Y','Trans Z','Rot X','Rot Y','Rot Z'})
xlabel('Axis')
ylabel('Activation Percentage')
title('Task 5')
hold on
sigstar(at5, p_at5)
errorbar(means(5,:), stds(5,:),'LineStyle','None')
hold off

sgtitle('Activation Percentage vs Axis for each Task')
```



Activation Percentage vs Task (Overlay)

```
figure(5)
plot(1:5, means)
title('Activation Percentage vs Task for all Axes')
axis([0.9 5.1 0 1])
xticks([1 2 3 4 5])
xticklabels({'1','2','3','4','5'})
xlabel('Task')
ylabel('Activation Percentage')
legend('Trans X','Trans Y', 'Trans Z', 'Rot X', 'Rot Y', 'Rot
 Z','Location', 'northeast','NumColumns',3)

% %% Check Some Random Stuff
%
% figure(6)
% hist(jaco_task1_vel(:,1),50)
```



Write Task Data into CSV Files

```
csvwrite('/home/tem/Documents/MATLAB/BMI/KD/CSV/jaco_task1.csv',
 jaco_task1_vel);
csvwrite('/home/tem/Documents/MATLAB/BMI/KD/CSV/jaco_task2.csv',
 jaco_task2_vel);
csvwrite('/home/tem/Documents/MATLAB/BMI/KD/CSV/jaco_task3.csv',
 jaco_task3_vel);
csvwrite('/home/tem/Documents/MATLAB/BMI/KD/CSV/jaco_task4.csv',
 jaco_task4_vel);
csvwrite('/home/tem/Documents/MATLAB/BMI/KD/CSV/jaco_task5.csv',
 jaco_task5_vel);
```

Published with MATLAB® R2019a