## Trajectory Preprocessing

All preprocessing was done according to Gallivan, J. P., & Chapman, C. S. (2014). Three-dimensional reach trajectories

### Fill Missing Data

Applies only to trajectories that have more than 1 sample.

Replaces NaNs with interpolated values.

Uses inpaint\_nans to do so.

**Inpaint\_nans**:

Written by John D'Errico.

Citation:

John D'Errico (2022). inpaint\_nans (https://www.mathworks.com/matlabcentral/fileexchange/4551-inpaint\_nans), MATLAB Central File Exchange. Retrieved February 15, 2022.

Is a advanced method for interpolating trajectory data while keeping the dynamic information of the reach.

Was used by Gallivan, J. P., & Chapman, C. S. (2014). Three-dimensional reach trajectories

### Filter Trajectory

Applies only to trajectories that have more than 1 sample.

Apply low pass butter worth filter of 2nd order with cutoff at 8Hz on each axis separately to clean any noise in the recording.

Since the noise is canceled, we can later overfit the data without concern for capturing noise with our fit (Using Functional Data Analysis\_v1\_april2011).

### Set Origin

Set the first sample in each trial as the axes origin by reducing it from each sample.

Set the time of the first sample in each trial as t=0 by reducing it from each sample.

### Trim Onset Offset

Applies only to trajectories that have more than 1 sample.

Compute velocity and apply low pass butter worth filter of 2nd order to it with a 10Hz cutoff.

**GetOnset:**

Trim trajectory to the first sample in which the 3 consecutive samples had a velocity greater than 2mm/s and a total acceleration greater than 2mm/s^2.

**GetOffset:**

Trim trajectory ending as the point closest to the screen.

### Normalize

We wish to compare the X coordinates between trials. To do so the Z index must be equal between trials.

We achieve that by fitting a function to the data and extracting 200 points that are equally spaced along the Z trajectory.

Applies only to trajectories that have more than 2 samples.

toNormalize = I think this is the marker you wish to normalize (it is an input to the normalization function).

**B-spline –** is a function composed of sections (splines), each section is a separate polynomial function.

Degree – the highest order of the polynomials.

Knot – the point where two splines connect.

Knots = coefficients + degree.

B – basis functions. A list of functions that are used to create each of the spline functions.

Cubic spline – spline function where all the polynomials are of the 3rd degree.

C – Continuity, a property of a spline indicating how smooth it is.

**Roughness penalty –** A parameterthat is added to the spline function and determines how

much emphasis to put on smoothness (as opposed to overfitting the data).

This type of spline is called a "Smoothing spline".

Error function (which we try to minimize when fitting the spline):

Text, whiteboard

Description automatically generated

– penalty factor, determines how much weight will be given to smoothness as opposed to

overfitting the data.

~~Chart, diagram

Description automatically generated~~

More information about roughness penalty might be in these links:

[Mathlab's cubic spline function](https://www.mathworks.com/help/curvefit/csaps.html#d123e27998)

[Article about functional data analysis](https://www.researchgate.net/publication/227578062_Functional_Data_Analysis)

[R Function for fitting splines](https://link.springer.com/content/pdf/10.1007%2F978-0-387-98185-7_5.pdf)

Fit a B-spline function to the each axis with a spline at each data point. The B-spline is of the 6th order which keeps the original data up to the 3rd derivative (Using Functional Data Analysis\_v1\_april2011).

A roughness penalty is applied to the 4th derivative of the data to smooth the data.

Use the fitted function to produce a high-resolution representation of the trajectory (1000 samples).

Perform the normalization in space by extracting 200 points equally spaced along the Z axis (perpendicular to the screen) from the high-resolution representations.

Since the subject can move backwards the movement is divided to sections according to the movement direction (towards / away from the screen). The 200 points are equally divided along the TOTAL path traveled in the Z axis (if the subject moved 2 forward, 1 backward and 3 forward, the total is 6).

## Trial screening

Test the original data (before preprocessing) and excludes trials that fall within the following criteria:

* Too early – movement initiation started lees than 100ms after target was displayed indicating a predictive response.
* Late response – movement didn't start 320ms after target display.
* Extremely slow movement – movement time was 3SD above the subject's mean, or simply above a preset threshold.
* Short reach distance – distance along the Z axis between first and last sample was shorter than 32cm.
* Too much missing samples – trajectory has more than 100ms of missing samples (usually due to recording issues or obstruction of the marker from the camera's view).
* Short sample – Recorded trajectory is shorter than 100ms.
* Missed targets – The last sample is further than 12cm from either of the targets (on the X,Y plane).
* Bad stimulus duration – the duration of one of the stimuli (fixation, masks, prime, target) deviated by more than 2ms from its designated duration.
* Incorrect – target classification was wrong.
* Subject didn't perform the trials (quit)

## Subject screening

Excludes subjects that fall withing the following criteria:

* Not enough trials – The number of valid trials which also had a PAS rating of 1 is smaller than 60.
* Not enough trials in each condition – The number of valid trials in each condition (congruent /

incongruent) which also had a PAS rating of 1 is smaller than 30.

* Bad performance – subject was correct less than 70% of the times in the target classification task (counts only trials with PAS=1).
* Aware of prime – prime recognition was above chance indicating the subject is aware of the prime.

## Parameters extraction

### MAD

Maximal absolute deviation.

Locates for each trial the point that is furthest away from the line that connects the start and end points (known as the maximally deviating point, mad\_p).

MAD is the distance of that point from that line.

### Functional data analysis

getRMMeans – calculates the mean trajectory for each subject at each condition.

Fanovan – runs a repeated measures aNOVA analysis at every point along the trajectory.

For an explanation why this doesn't violate multiple comparisons limitations see Craig's document " Using Functional Data Analysis\_v1\_april2011.pdf" section: 5.2 – Statistical violation and functional interpretation.

Main point is that we conclude that the two trajectories are different from finding a significant difference at some point along the trajectory, but we do not draw conclusions about single points along the trajectory.

Subject and left/right are used as random factors.

### Reach area calculation

Calculates the area circumscribed between the average trajectory to the left target and the average trajectory to the right target.

Does so for each condition separately.

Area calculation:

~~Trim one trajectory's Z values to be within the range of the other trajectory.~~ Isn't relevant since they share Z values.

Finds the minimal X value and draws a line at that value parallel to Z axis.

Computes the area between each trajectory and this line and subtracts the results, thus receiving the area between the trajectories.

To avoid negative area values after the trajectories intersect, they are split to section at their intersections and the area is calculated separately for each section.

## Averaging within subject

Averages trials withing each participant, excluding invalid trials or trials whose PAS isn't 1.

Averages:

* Trajectories
* Response time
* Reaction time
* Movement time
* Prime recognition – Includes most invalid trials, \*Excludes only\* unperformed trials or trials with bad

stimuli duration.

* MAD
* Maximally deviating point.
* Total STD of the X coordinates – collapses the STD across time.

Computes:

* STD (between trials) of the x coordinates at each point along Z.
* Number of trials with each PAS rating.
* Difference between average trajectory in the congruent and incongruent conditions.

## Sorting and averaging between subjects

Averages the following values (after excluding the bad subjects):

* Trajectory
* RT
* Reaction time
* Movement time
* Prime recognition
* Number of trials with each PAS rating.
* MAD
* Maximally deviating point.
* Reach area
* STD of the x coordinates at each point along Z.
* Total STD of X

## Formatting to R

### Reach area

Define 'M', which is the minimal amount of valid trials any subject has in a single condition.

For each subject we compute 1000 average left and right trajectories for each condition by randomly sampling (with replacement) 'M' trials and averaging them.

We use these average trajectories to compute 1000 reach areas for each subject in each condition.

## Modeling

### Diff in X value between conditions along the trajectory

We computed the difference between each subject's average congruent and incongruent trajectories in the "averaging within subject" section (Did so for left and right sides separately).

We now plot the average difference and it's confidence interval, this gives us a "t-test" along time. We look if at any point in time the difference is significantly greater than zero.

Chart

Description automatically generated

Chart, surface chart

Description automatically generated

### Reach Area

We previously generated 1000 reach area observations for each subject.

We enter all these observations from all the subject into a LMM with condition as the predictor, reach area as the predicted, and subject as a random factor.

We then compare it to an "empty" model that contains only an intercept and subjects as a random factor.

Text

Description automatically generated with low confidence

### MAD

Model MAD as a function of condition with subjects as a random factor.

Use actual values for each subject, not the average MAD.

A screenshot of a computer

Description automatically generated with medium confidence

### X Deviation

Model X position as a function of condition with subjects as a random factor.

Does so for each point along the trajectory.

A screen shot of a computer

Description automatically generated with low confidence

## ניתוח

* מספר הפעמים שנבדק שינה את דעתו במסלול

מדד לאפקט.

Resulaj עשה את זה.



* ניתוח זווית (כיוון) בכל נקודה בזמן.



### Linear mixed model

#### Reach area

נראה שה-interpolation של ה-trajectories לא עובדת טוב מה שגורם למסלולים מעוותים.

אדום = הושטה שמאלה (כן זה הפוך), ירוק = הושטה ימינה.

Chart, line chart

Description automatically generated

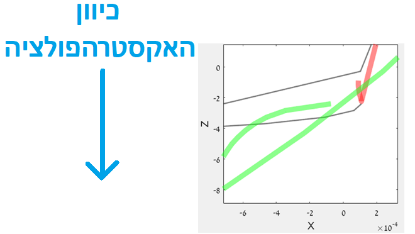
זה יוצר ערכים קיצוניים ב-reach area.

Chart, waterfall chart

Description automatically generated

מבט מקרוב מגלה שהמסלול האדום מתחיל בנקודה מאוחרת על ציר מהמסלול הירוק, משמע שצריך לעשות לו אקסטרהפולציה כדי לקבל את ערכי X של המסלול האדום בכל אחת מנקודות ה-Z של המסלול הירוק.

הבעיה היא שהמסלול האדום לא פונה לכיוון אליו רוצים לעשות אקסטרה פולציה, ולכן ה-fit צריך להתאים ערכים קיצוניים כדי לנסות לפצות על כך.



כדי לפתור זאת נציב סף על ציר Z, ונחשב את השטח רק מעבר לסף זה.

כדי להימנע מביקורת עתידית, ניתן להכניס רעש לתחילת המסלול ולהראות ששינוי הסף אינו יוצר אפקט בשל הרעש.