Internship Report
at Decision, Action, and Neural Computation (DANC) laboratory in Institut des Sciences Cognitives(ISC), CNRS, Lyon

under guidance of
Prof. James Bonaiuto

Title - Decoding real and imagined movement from beta burst dynamics using deep neural networks

by

Tanisha Pal,
MS by Research, 1st year at
International Institute of Information Technology(IIIT) Hyderabad, India

May 2025 to July 2025

1. Experiment Description:

Thirty-eight healthy, right-handed adults (25 females, aged 20–35 years, M = 26.69, SD = 4.11) took part in this visuomotor adaptation study. All participants had normal or corrected-to-normal vision, provided written informed consent, and were approved by the regional ethics committee (CPP Est IV - 2019-A01604–53).
Participants performed a joystick-based reaching task, where they had to quickly move a cursor to a visually indicated target on a screen. At the start of each trial, they fixated on a central symbol. A random dot motion (RDK) stimulus then appeared, showing dots moving either clockwise, counterclockwise, or randomly. Around the RDK, five possible reach targets appeared, and one was highlighted in green, indicating the target they had to reach after a go cue.
The key manipulation was the visuomotor rotation, where the position of the cursor was rotated by ±30° or 0°, depending on group and trial. Participants were divided into two groups:
Explicit group (N = 20): The cursor rotation varied with the direction of RDK motion. This allowed participants to consciously anticipate and correct for the rotation using the visual motion cue.

- Rotation direction (±30°, 0°) was predictable based on the direction of RDK motion.
- Participants could consciously adjust their movements based on the visual motion cue.
- Example: If dots moved clockwise, they could expect a +30° rotation and plan accordingly.

Implicit group (N = 18): The cursor was always rotated by −30°, regardless of the RDK motion. Participants could not predict the rotation but could gradually adapt to it over time without being explicitly aware of it.

- Rotation was always −30°, and unrelated to the RDK motion.
- Participants could not rely on visual cues and had to gradually adapt to the constant rotation through repeated experience.
- Learning happened implicitly, without conscious awareness of the rotation rules.

Dot motion coherence (how many dots moved in the same direction) varied across trials, and each participant's sensitivity level was calibrated individually using a staircase procedure. Participants first completed training blocks with no motion or rotation. Then, they completed multiple blocks where the rotation and RDK coherence levels varied as described. The experiment ended with a "washout" block, with no motion or rotation.

In this study, neural activity was recorded using a 275-channel MEG system with SQUID-based axial gradiometers, which capture magnetic fields generated by neural currents. MEG was used to detect transient beta bursts (13–30 Hz) in the sensorimotor cortex during a joystick-based visuomotor task. These bursts, rather than being continuous oscillations, were identified as brief, diverse waveform events reflecting movement-related cortical dynamics. The study highlights that MEG captures heterogeneous beta burst motifs shaped by varying synaptic inputs to deep and superficial cortical layers, offering insights into distinct motor-related neural computations.

2. Dataset Overview:

- Total subjects: 38 (named sub-101, sub-102, ..., sub-138)

Behavioral CSV files: 9 files (one per block)

- sub-101-001-beh.csv (training block)
- sub-101-002-beh.csv to sub-101-009-beh.csv (8 experimental blocks)
- Each experimental block has 56 trials
- **Variables** – [subject_id, group, block, trial_in_block, trial_coherence, trial_perturb, trial_type, reach_dur, reach_rt, trial_directions, trial_target, aim_target, reach_target, aim_real_angle, reach_real_angle, true_target_angle, coh_cat, perturb_cat]

  Description of some of the important variables:
  - block – Identifies the trial block number within the experiment.
  - trial_in_block – Index of the trial within the specific block.
  - trial_target – The actual instructed visual target for the participant to reach.
  - aim_target – The target location aimed at by the participant.
  - reach_target – The final target reached by the joystick cursor.
  - aim_real_angle – Angular direction of the participant's intended reach.
  - reach_real_angle – Actual angular direction of the executed reach movement.
  - true_target_angle – True angle of the instructed visual target.
  - reach_dur – Duration of the reaching movement after the go cue.
  - reach_rt – Reaction time from the go cue to the start of the reach movement.

Burst JSON files (stored in a bursts/ folder):

- 22 total JSON files:
- 11 motor burst files → not relevant to my task
- 11 visual burst files → relevant to my task
- Each JSON file corresponds to one MEG sensor
- **Variables** – [trial, waveform, peak_freq, peak_amp_iter, peak_amp_base, peak_time, peak_adjustment, fwhm_freq, fwhm_time, polarity, waveform_times, block, pp_ix]

  Description of some of the important variables:
  - peak_freq – Frequency at which the burst has its maximum amplitude.
  - peak_amp_iter – Amplitude of the burst at its detected peak during iterative detection.
  - peak_amp_base – Peak amplitude relative to baseline power spectrum.
  - peak_time – Time point in the trial where the burst reaches its maximum amplitude.
  - peak_adjustment – Time adjustment made to align the waveform peak to zero-phase point.
  - fwhm_freq – Full width at half maximum in the frequency domain (i.e., frequency span).
  - fwhm_time – Full width at half maximum in the time domain (i.e., duration of burst).

3. Data pre-processing:

We first split the main behavioral dataset into individual subject-wise CSV files. For each subject, we loaded their behavioral CSV and merged it with the corresponding burst (JSON-derived) data using block and trial_in_block columns.
This resulted in a new merged CSV file per subject with **35 columns**, saved for downstream analysis.

1. For **Subject 101**, there were **9 behavioral CSVs**, one of which was a training block and excluded. The remaining **8 blocks** had **56 trials each**, resulting in **448 rows and 18 columns** of behavioral data.

- The burst data came from **11 JSON files**, each converted to CSV and then merged. The final merged burst file for Subject 101 had **237,372 rows and 13 columns**.
- This size comes from: **8 blocks × 56 trials/block = 448 trials**, and each trial had ~50 bursts, totaling about **22,400 bursts**.
- With **11 MEG sensors**, the final size became approximately **22,400 × 11 ≈ 240,000 rows**.
- Finally, the behavioral and burst data were aligned on a per-trial basis, resulting in the final merged file of size **237,372 rows × 31 columns**.

Next we merged, we added five key behavioral variables to the burst data: **reach_vis_err**, **reach_vis_abs_err**, **aim_vis_err**, **aim_vis_abs_err**, and **aim_vis_err**. This resulted in a new merged CSV file for each subject with 35 columns, saved for further analysis.

3.1 Data Cleaning for Behavioral Variable Prediction:

To ensure accurate modeling and avoid the influence of outliers or noisy data, we applied a filtering step to exclude behaviorally implausible trials. Specifically, we omitted any rows where the values of the four key behavioral variables — **reach visual error**, **aim visual error**, **reach visual absolute error**, and **aim visual absolute error** — fell outside the physiologically reasonable range.
We retained only those trials where:

- reach_vis_err and aim_vis_err were between **-90° and 90°**,
- reach_vis_abs_err and aim_vis_abs_err were between **0° and 90°**.

This filtering ensures that all input data used in our prediction models corresponds to valid motor behaviors, reducing noise and improving the reliability of both classification and regression results. After applying these filters, we proceeded to construct the final sensor-aware, time-binned burst matrices for each trial to be used in Transformer-based models.

4.Data-Processing

To perform Transformer-based classification to distinguish between trial targets 1 and 3 using burst-aligned waveform data. Following this, we predict six key behavioral variables — reach duration, reach reaction time, reach visual error, reach visual absolute error, aim visual error and aim visual absolute error — using regression models based on the same input features. This dual approach enables both categorical decoding of trial type and continuous estimation of visuomotor behavior from neural burst signatures.

We chose **Transformers** for both classification and prediction tasks due to their ability to effectively model long-range temporal dependencies in waveform data using self-

attention. Unlike traditional models, Transformers can focus on the most informative time points across trials, making them well-suited for learning subtle patterns in EEG/MEG bursts. Their scalability and interpretability through attention maps further enhance both performance and analysis.

4.1 Transformer Model for Classification

The model is designed to classify waveform inputs into two target classes using a Transformer-based architecture optimized for temporal sequence learning. The architecture consists of the following key components:

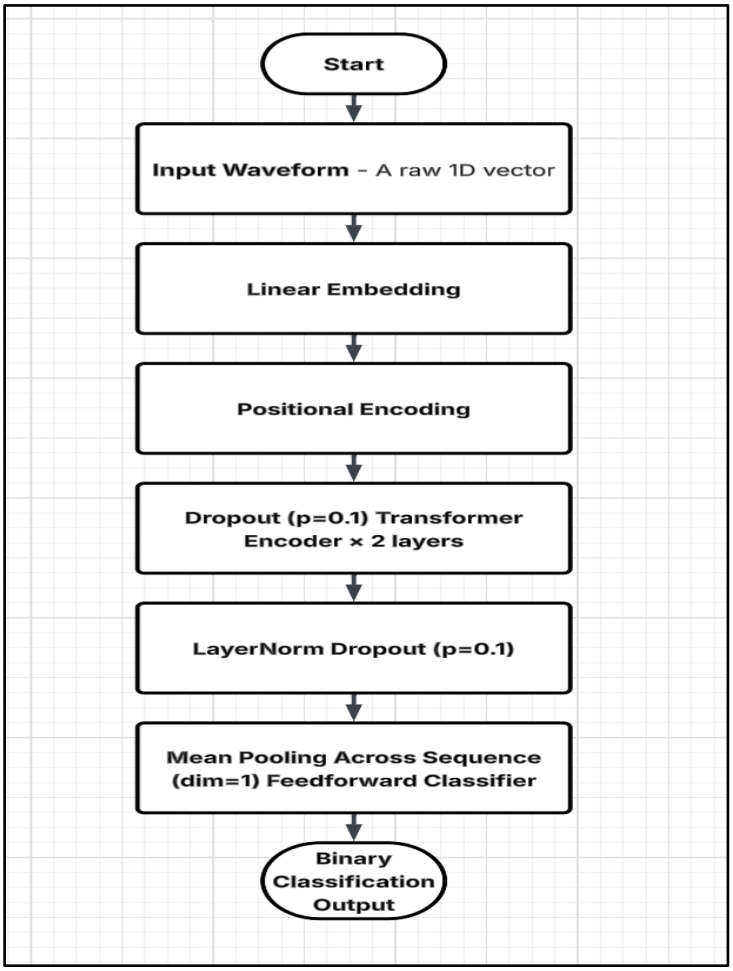
1. Input Layer
   - Input: A 1D waveform vector of shape [batch_size, input_dim]
   - input_dim: Number of input features (e.g., 156)
2. Linear Embedding
   - Projects input to a higher-dimensional space: nn.Linear(input_dim, hidden_dim)
   - Output shape: [batch_size, 1, hidden_dim], hidden_dim = 64
3. Dropout (p = 0.1)
   - Regularization applied after embedding
4. Positional Encoding
   - Learnable parameter added to the embedded input to retain temporal information: self.positional_encoding[:, :x.size(1), :]
5. Transformer Encoder
   - Built using nn.TransformerEncoderLayer with:
     - num_layers = 2
     - n_heads = 4
     - hidden_dim = 64
     - Dropout within attention and feedforward layers: p = 0.1
   - Captures contextual relationships in the embedded and position-encoded input
6. Layer Normalization
   - Normalizes the Transformer output to stabilize training: nn.LayerNorm(hidden_dim)
7. Dropout (p = 0.1)
   - Applied before classification
8. Sequence Pooling
   - Mean pooling across the sequence dimension to get a fixed-size representation: x = x.mean(dim=1)
9. Feedforward Classifier
   A two-layer fully connected network:
   - Linear(hidden_dim → hidden_dim//2)
   - ReLU activation
   - Dropout(p = 0.3)
   - Linear(hidden_dim//2 → 2) → Final logits for two classes
10. Output

- Final output shape: [batch_size, 2]
- Used with CrossEntropyLoss for training

4.1.1 Training Configuration

For the binary classification task, we employed a Transformer-based model capable of capturing temporal dependencies within burst waveforms. Through several rounds of hyperparameter tuning, we optimized the architecture by adjusting the number of Transformer encoder layers, attention heads, and dropout values. We compared different classifier heads and finalized a two-layer feedforward classifier with ReLU activation and dropout for regularization. The model was trained using the CrossEntropy loss and an Adam optimizer with a learning rate of 1e-4 and weight decay of 1e-5. This setup yielded consistent training and test accuracy across runs.

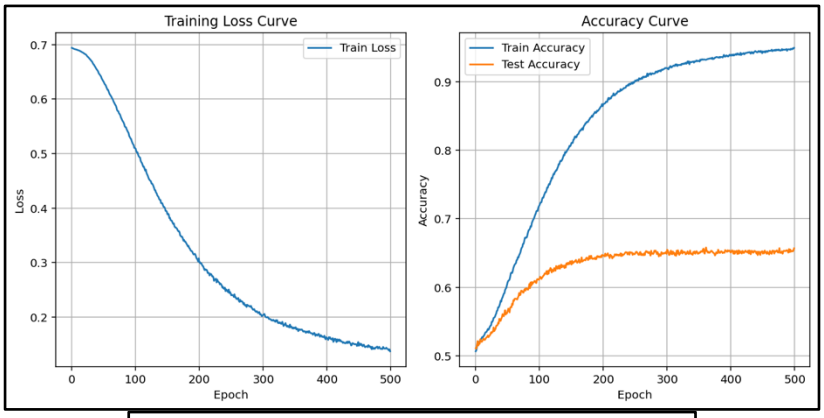
Model: Transformer (2 layers, 4 heads, hidden dim = 64)
Loss: CrossEntropyLoss
Optimizer: Adam (lr = 1e-4, weight decay = 1e-5)
Epochs: 500
Batch size: 64
Device: MPS > CUDA > CPU
Train/Test split: 80/20 with stratification

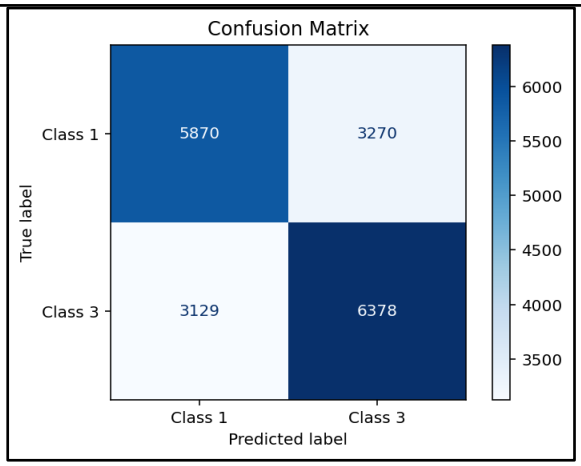


4.1.2 Results for Transformer based Classification

The Transformer classifier achieved high training accuracy (~94.7%–94.9%) in distinguishing between trial target 1 and trial target 3, indicating effective learning of the training data. However, the test accuracy plateaued around 65.6%, suggesting moderate generalization performance. This gap between training and test accuracy points to potential overfitting, and highlights the need for further regularization, data augmentation, or architectural tuning to improve generalization on unseen trials.

Epoch 491: Loss=0.1420, Train Acc=0.9469, Test Acc=0.6506
Epoch 492: Loss=0.1419, Train Acc=0.9470, Test Acc=0.6557
Epoch 493: Loss=0.1420, Train Acc=0.9468, Test Acc=0.6514
Epoch 494: Loss=0.1436, Train Acc=0.9462, Test Acc=0.6551
Epoch 495: Loss=0.1418, Train Acc=0.9477, Test Acc=0.6519
Epoch 496: Loss=0.1412, Train Acc=0.9469, Test Acc=0.6529
Epoch 497: Loss=0.1408, Train Acc=0.9470, Test Acc=0.6531
Epoch 498: Loss=0.1385, Train Acc=0.9485, Test Acc=0.6523
Epoch 499: Loss=0.1406, Train Acc=0.9478, Test Acc=0.6538
Epoch 500: Loss=0.1370, Train Acc=0.9491, Test Acc=0.6568





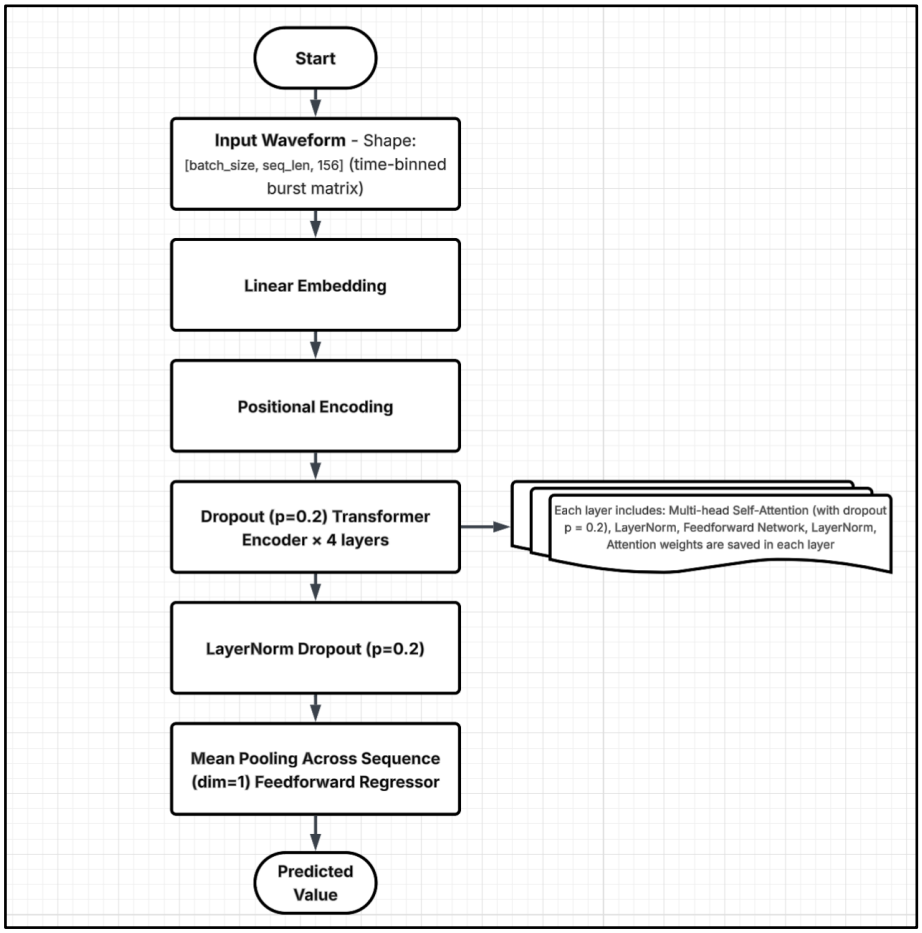
4.2 Transformer Model for Prediction

We group the data by trial and divide the burst waveforms into fixed time bins (50 ms each) based on their peak times.
The objective of this process is to achieve **burst synchronization across trials** by aligning burst waveforms into consistent time bins and sensor positions. This structured representation ensures that temporal and spatial burst patterns are comparable across trials, enabling the Transformer to effectively learn relationships between burst dynamics and behavioral outcomes.
For each time bin and each sensor, we compute an average waveform if any bursts exist; otherwise, we fill with zeros. These bin-wise waveforms across all sensors are arranged into a structured 3D tensor of shape (sensors, time_bins, 156) per trial. Each sensor-specific matrix is then appended to the dataset. This results in the final input matrix X of shape (num_trials × sensors, time_bins, 156) and the corresponding target variable vector y, ready for input into the Transformer regressor.

4.2.1 Transformer Model for Prediction – Architecture

This Transformer-based regression model is designed to predict behavioral variables from time-binned burst EEG features. Here's a detailed breakdown of the architecture:

1. Input
    a. Shape: (Batch, Sequence Length = 55, Feature Dimension = 156).
    b. This input represents burst-aligned waveforms binned over time and sensors.
2. Embedding Layer
    a. A fully connected (linear) layer projects each 156-dimensional input vector to a 64-dimensional hidden space.
    b. Layer: Linear(156 → 64)
3. Positional Encoding
    a. A learnable positional encoding of shape (1, 55, 64) is added to the embedded input to retain sequence order.
4. Transformer Encoder Layers
    a. Number of Layers: 4
    b. Each layer includes:
        i. Multi-head self-attention with 4 heads
        ii. Feedforward network: Linear(64 → 256 → 64)
        iii. Residual connections and Layer Normalization
        iv. Activation Function: ReLU
        v. Dropout Rate: 0.2 (applied after attention and feedforward)
5. Layer Normalization + Dropout - A final LayerNorm and Dropout(0.2) is applied after all Transformer blocks to stabilize and regularize the output.
6. Regression Head
    a. The output is mean-pooled over the sequence dimension.
    b. Then passed through:
        i. Linear(64 → 32) with ReLU and Dropout(0.4)
        ii. Linear(32 → 1) for final prediction



4.2.2 Training Configuration

For the regression task, we adapted the Transformer to predict continuous behavioral variables from time-binned burst matrices. We evaluated both MSE and Huber loss, ultimately selecting MSE due to its better performance on our data. We explored various architectural configurations—varying the number of Transformer layers (up to 4), attention heads, hidden dimensions, and dropout rates. The final model incorporated learnable positional encoding, mean pooling across time, and a regularized feedforward regressor. With a learning rate of 1e-4 and weight decay of 1e-5, this configuration achieved optimal validation performance with stable and interpretable output.

- Loss Function: Mean Squared Error (MSE)
- Optimizer: Adam
- Learning Rate: 1e-4
- Weight Decay: 1e-5
- Epochs: 1000
- Batch Size: 64
- Train/Test Split: 80/20
- Normalization:
    - Input features (X) normalized per feature

- Output target (y) normalized using StandardScaler and then inverse-transformed for evaluation

Prediction was performed on 2 sets:

1. On Individual Subject
2. On 2 groups – Implicit and Explicit

4.2.3 Results

1. **On Individual Subject** – After getting the R2 score for the individual subjects – few subjects perform very low – 112,117,132,140,141,145. Their scores were much below 0.4 while predicting all the 6 variables – so we removed them.

Results on Individual subjects –

I. https://github.com/danclab/burst_transformer/blob/main/analysis_plotsv2_aim_vis_abs_err.pdf
II. https://github.com/danclab/burst_transformer/blob/main/analysis_plotsv2_aim_vis_err.pdf
III. https://github.com/danclab/burst_transformer/blob/main/analysis_plotsv2_reach_dur.pdf
IV. https://github.com/danclab/burst_transformer/blob/main/analysis_plotsv2_reach_rt.pdf
V. https://github.com/danclab/burst_transformer/blob/main/analysis_plotsv2_reach_vis_abs_err.pdf
VI. https://github.com/danclab/burst_transformer/blob/main/analysis_plotsv2_reach_vis_err.pdf

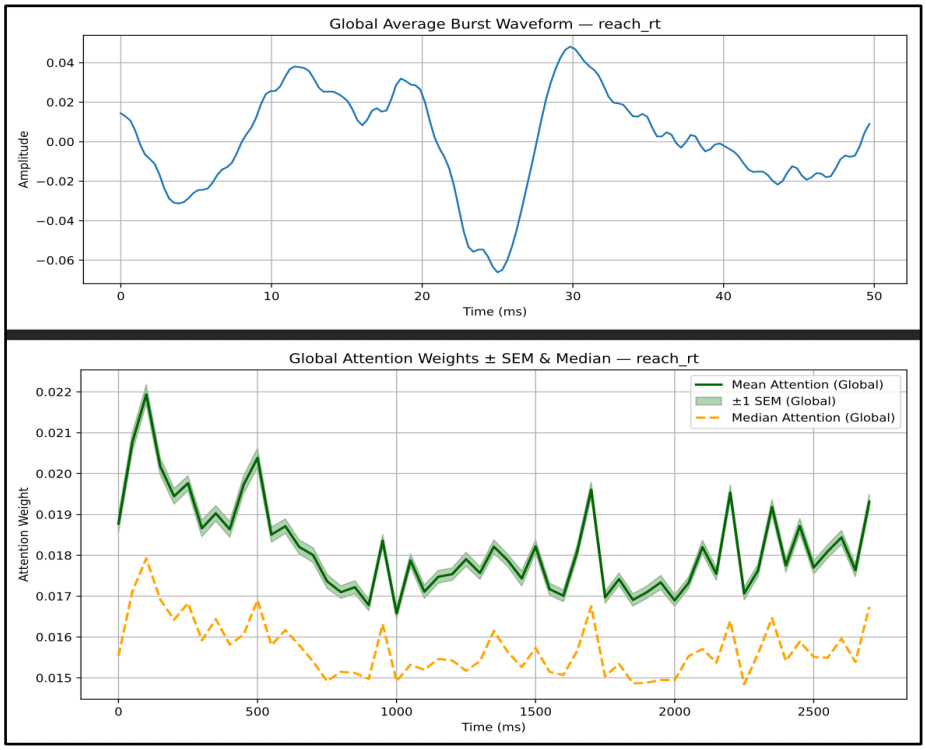
a. Plot Descriptions

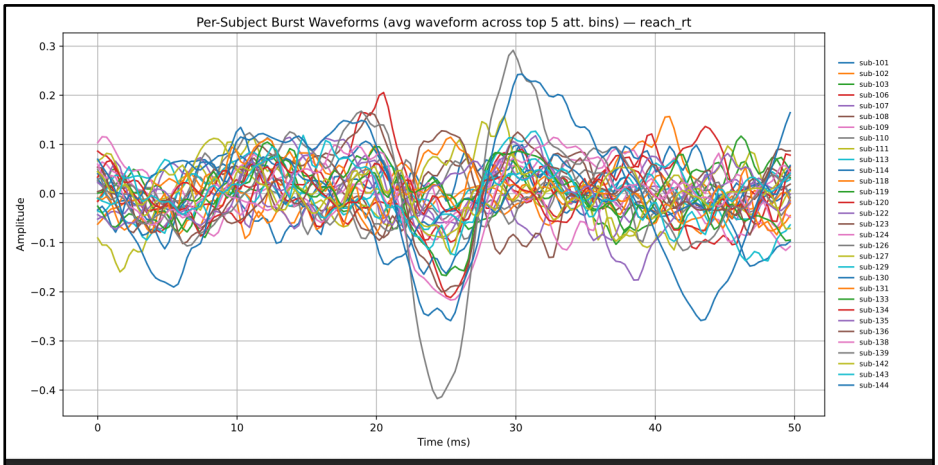
i. Global Average Burst Waveform
   1. This plot shows the average waveform shape across all subjects.
   2. It is calculated by averaging burst waveforms (length 156) across all subjects.
   3. The x-axis represents time (ms) and the y-axis shows amplitude.
   4. This gives a general picture of how a typical burst looks across the population.

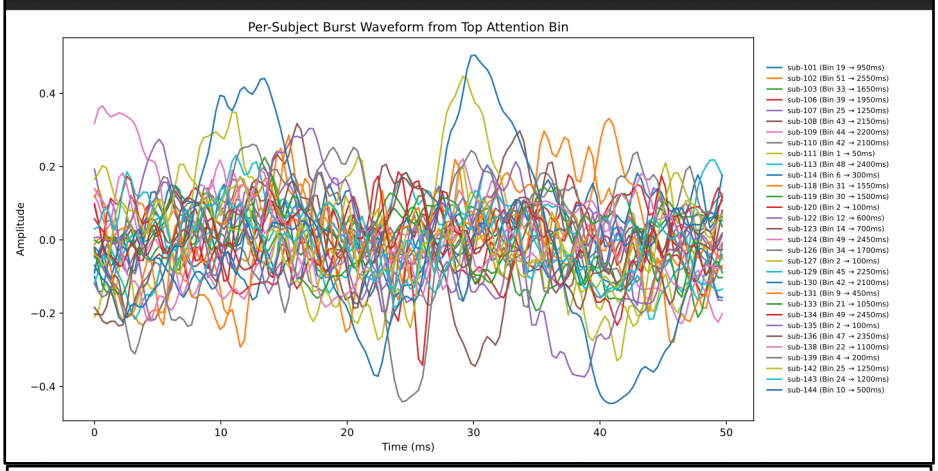
ii. Global Attention Weights (Mean ± SEM and Median)
   1. This plot visualizes attention weights over time for all subjects combined.
   2. Three curves are shown:
      a. Mean attention weights (solid green)
      b. Standard Error of the Mean (SEM) as a shaded region
      c. Median attention (dashed orange)
   3. The x-axis is time in milliseconds (based on bin index), and y-axis is attention weight.
   4. Helps identify the time periods that are globally most informative.

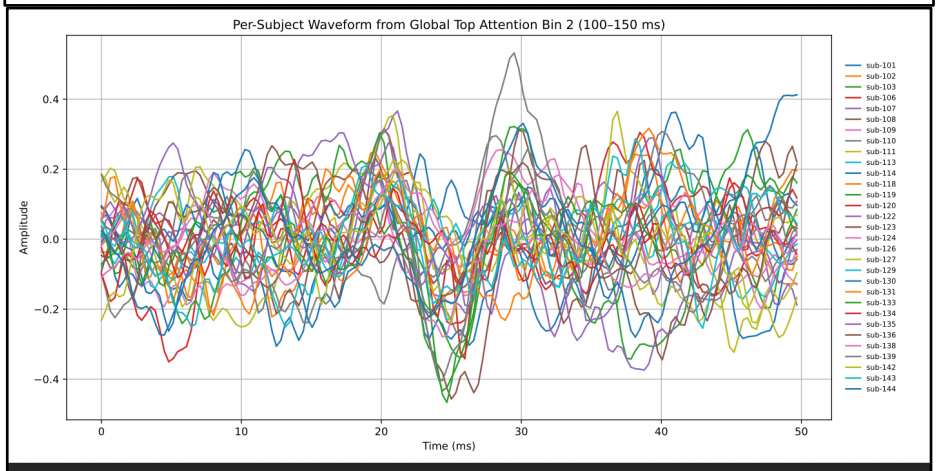
iii. Per-Subject Burst Waveforms (Avg across Top 5 Attention Bins)
    1. Each subject's average burst waveform is shown.
    2. These averages are computed using the top 5 attention-weighted time bins per subject.
    3. Helps compare individual waveform shapes across subjects.
iv. Per-Subject Burst Waveform from Top Attention Bin
    1. For each subject, the burst waveform from their most attended bin (highest attention weight) is plotted.
    2. Shows which time bin is most important for each subject and the corresponding average waveform.
    3. Labels include the subject ID and time (in ms) of that top bin.
v. Per-Subject Waveform from Global Top Attention Bin
    1. Uses the globally most attended time bin (highest global attention weight).
    2. For each subject, the average waveform from that same bin is shown.
    3. Allows direct comparison across subjects in terms of how their bursts behave at this key moment.
vi. Per-Subject Attention Weights
    1. Each line shows the mean attention weight over time for one subject.
    2. Helps visualize subject-wise variability in attention distribution across time bins.
    3. Useful to see whether some subjects focus on different time periods than others.

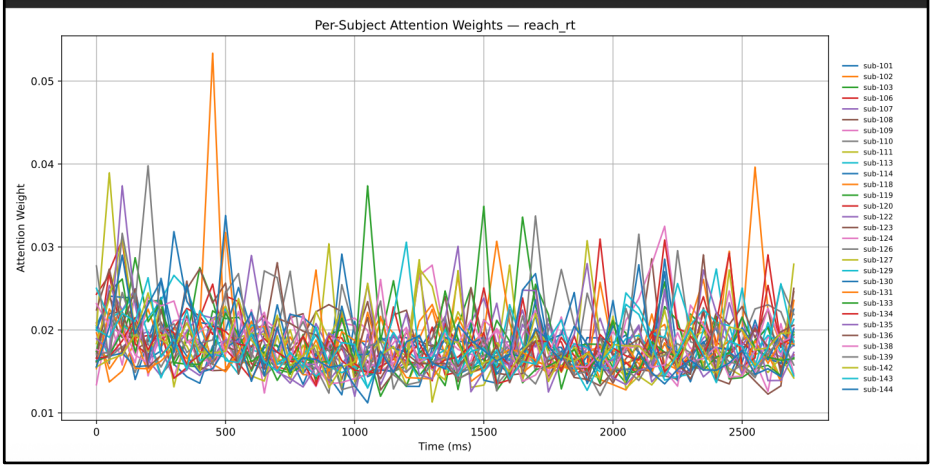
For the variable Reach_rt



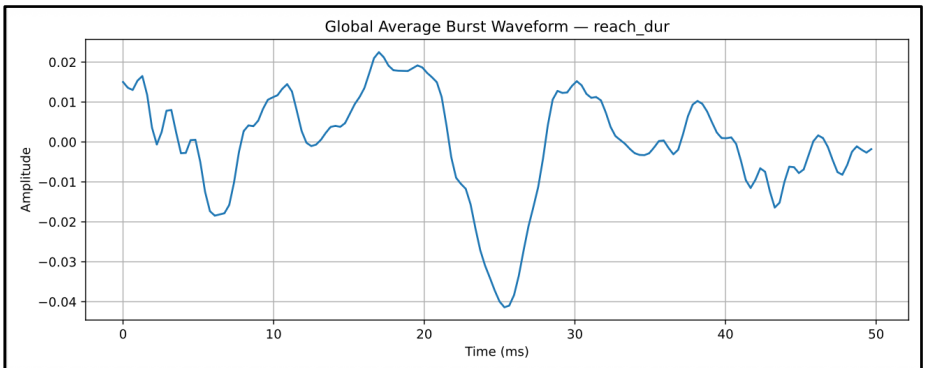

Per-Subject Burst Waveforms (avg waveform across top 5 att. bins) — reach_rt
Amplitude
Time (ms)
sub-101
sub-102
sub-103
sub-106
sub-107
sub-108
sub-109
sub-110
sub-111
sub-113
sub-114
sub-118
sub-119
sub-120
sub-122
sub-123
sub-124
sub-126
sub-127
sub-129
sub-130
sub-131
sub-133
sub-134
sub-135
sub-136
sub-138
sub-139
sub-142
sub-143
sub-144

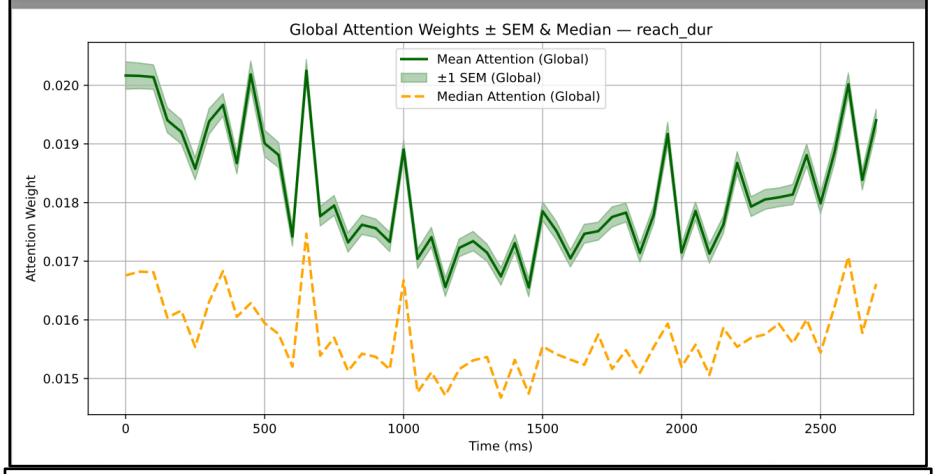

Per-Subject Burst Waveform from Top Attention Bin
Amplitude
Time (ms)
sub-101 (Bin 19 → 950ms)
sub-102 (Bin 51 → 2550ms)
sub-103 (Bin 33 → 1650ms)
sub-106 (Bin 39 → 1950ms)
sub-107 (Bin 25 → 1250ms)
sub-108 (Bin 43 → 2150ms)
sub-109 (Bin 44 → 2200ms)
sub-110 (Bin 42 → 2100ms)
sub-111 (Bin 1 → 50ms)
sub-113 (Bin 48 → 2400ms)
sub-114 (Bin 6 → 300ms)
sub-118 (Bin 31 → 1550ms)
sub-119 (Bin 30 → 1500ms)
sub-120 (Bin 2 → 100ms)
sub-122 (Bin 12 → 600ms)
sub-123 (Bin 14 → 700ms)
sub-124 (Bin 49 → 2450ms)
sub-126 (Bin 34 → 1700ms)
sub-127 (Bin 2 → 100ms)
sub-129 (Bin 45 → 2250ms)
sub-130 (Bin 42 → 2100ms)
sub-131 (Bin 9 → 450ms)
sub-133 (Bin 21 → 1050ms)
sub-134 (Bin 49 → 2450ms)
sub-135 (Bin 2 → 100ms)
sub-136 (Bin 47 → 2350ms)
sub-138 (Bin 22 → 1100ms)
sub-139 (Bin 4 → 200ms)
sub-142 (Bin 25 → 1250ms)
sub-143 (Bin 24 → 1200ms)
sub-144 (Bin 10 → 500ms)

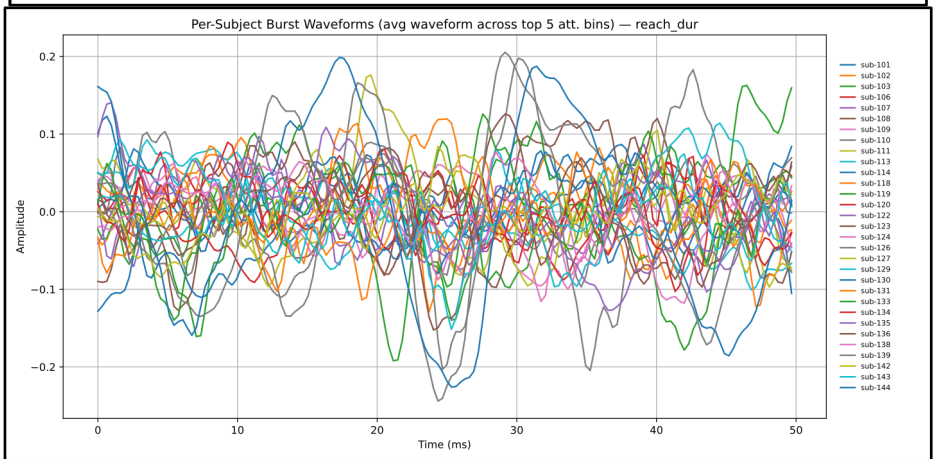

Per-Subject Waveform from Global Top Attention Bin 2 (100–150 ms)
Amplitude
Time (ms)
sub-101
sub-102
sub-103
sub-106
sub-107
sub-108
sub-109
sub-110
sub-111
sub-113
sub-114
sub-118
sub-119
sub-120
sub-122
sub-123
sub-124
sub-126
sub-127
sub-129
sub-130
sub-131
sub-133
sub-134
sub-135
sub-136
sub-138
sub-139
sub-142
sub-143
sub-144

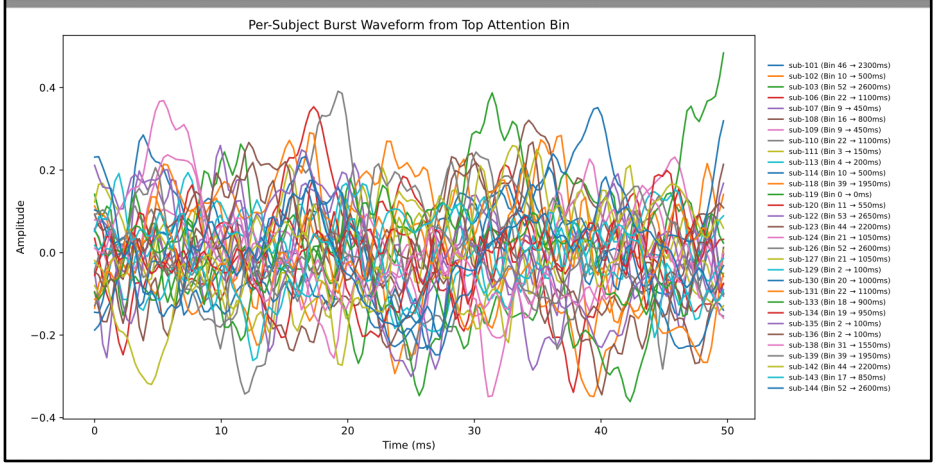

Per-Subject Attention Weights — reach_rt
Attention Weight
Time (ms)
sub-101
sub-102
sub-103
sub-106
sub-107
sub-108
sub-109
sub-110
sub-111
sub-113
sub-114
sub-118
sub-119
sub-120
sub-122
sub-123
sub-124
sub-126
sub-127
sub-129
sub-130
sub-131
sub-133
sub-134
sub-135
sub-136
sub-138
sub-139
sub-142
sub-143
sub-144

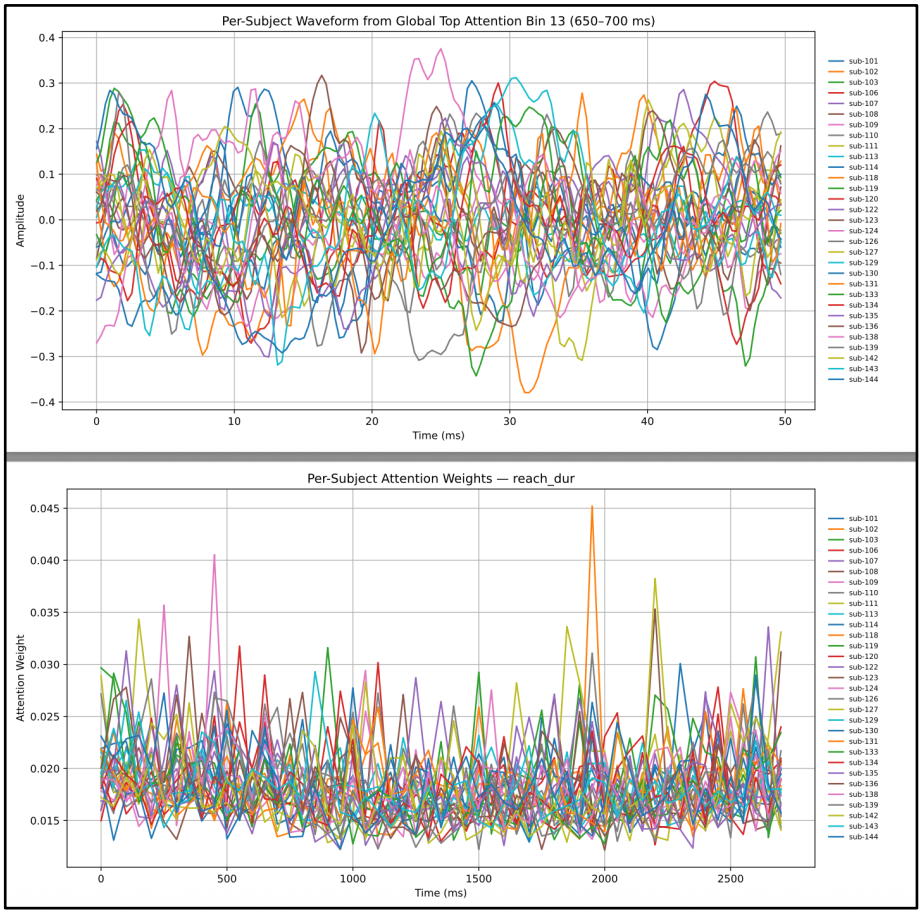
For the variable Reach_dur



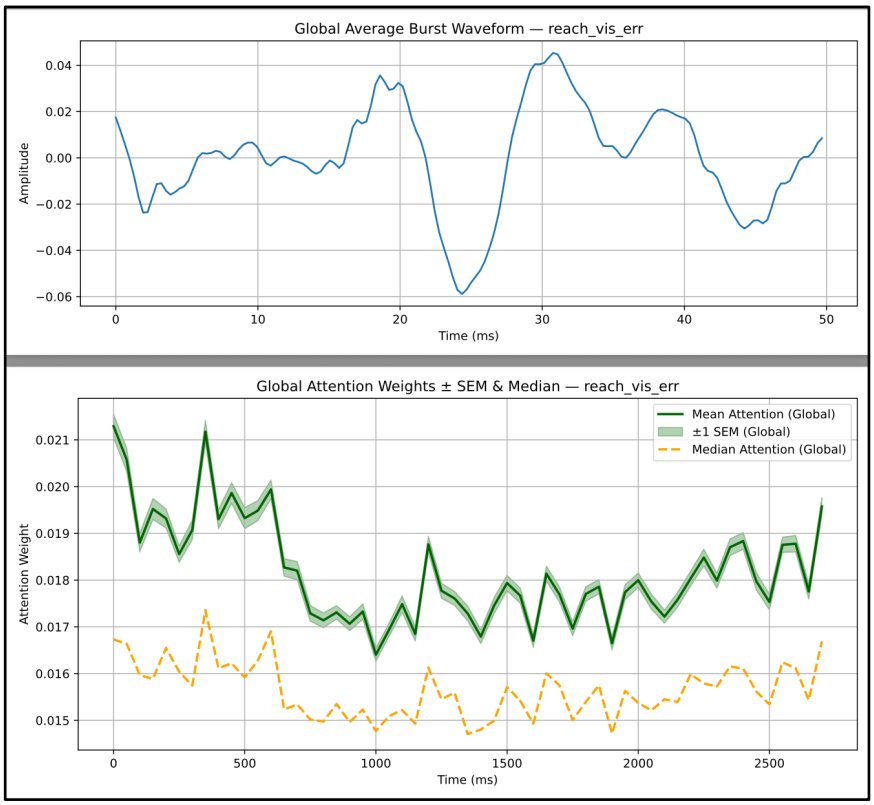


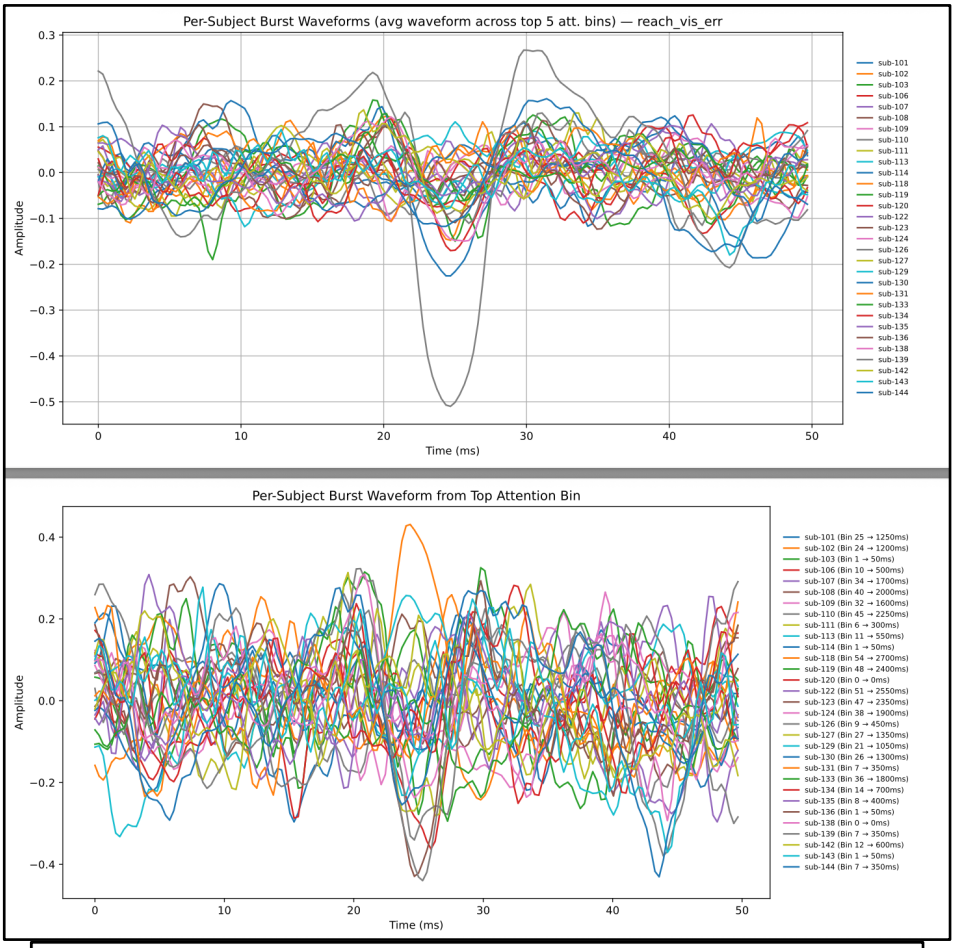




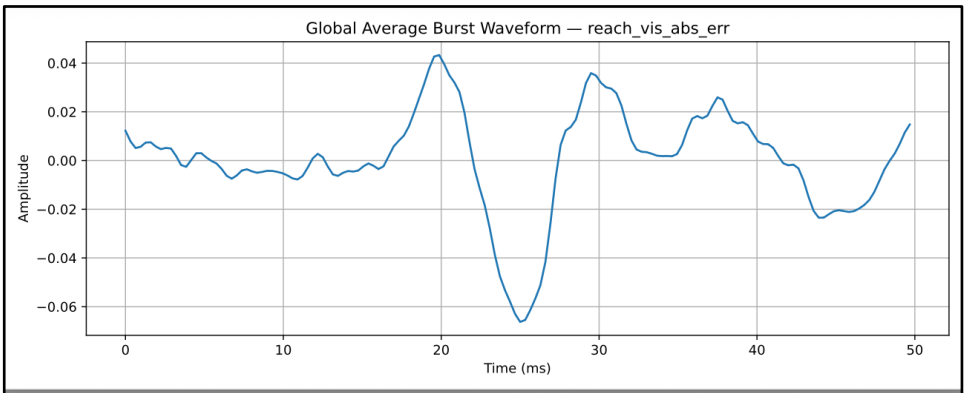


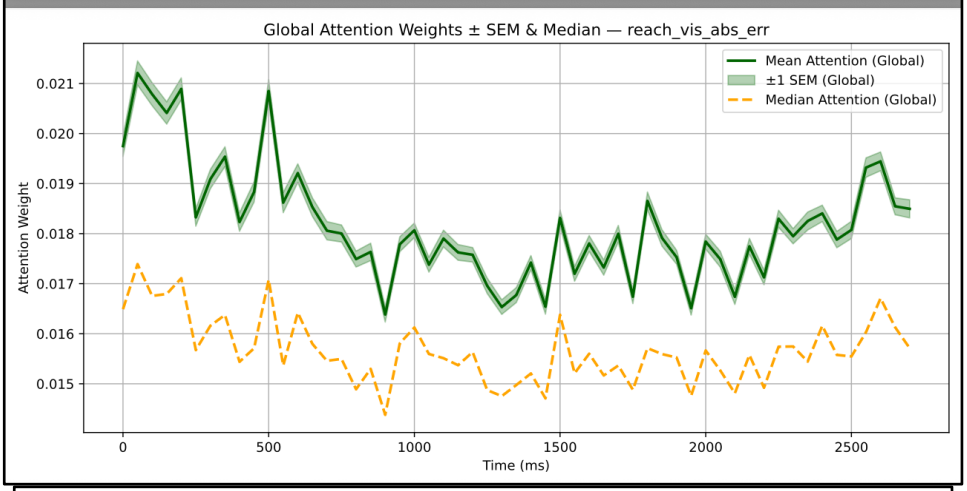
For the variable Reach_vis_err

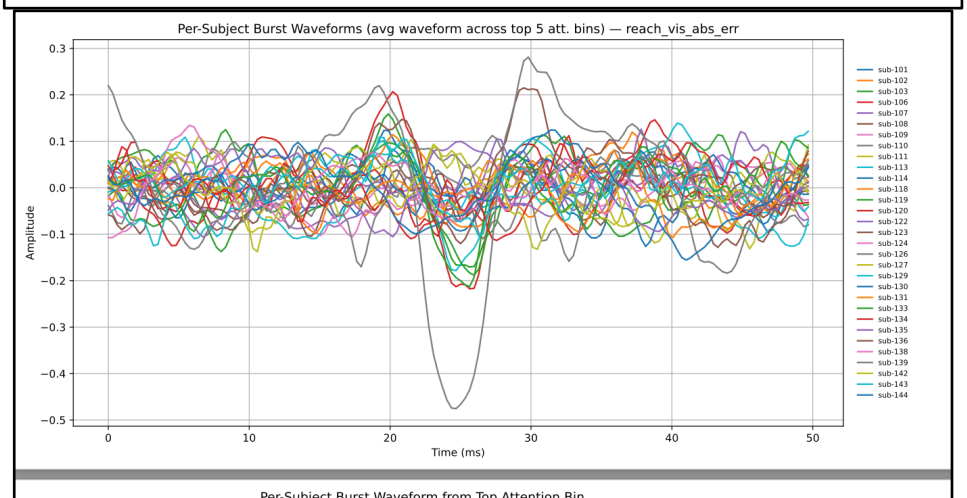


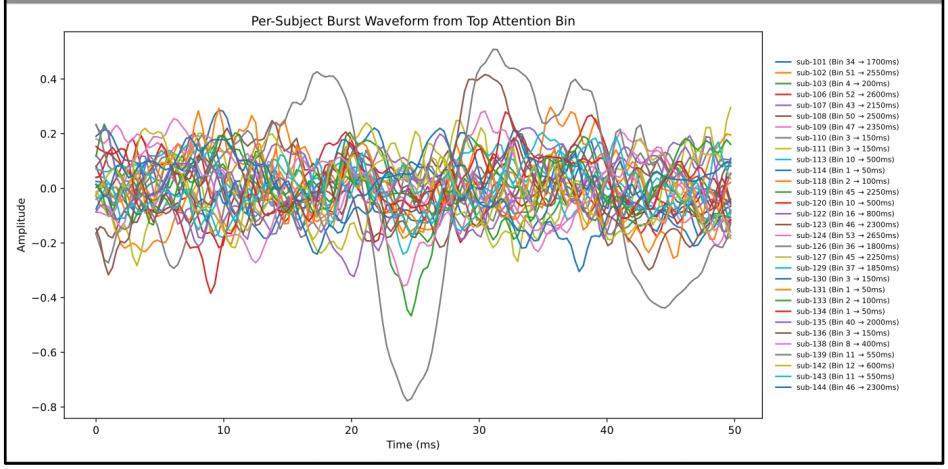

Per-Subject Burst Waveforms (avg waveform across top 5 att. bins) — reach_vis_err
Amplitude
Time (ms)
Per-Subject Burst Waveform from Top Attention Bin
Amplitude
Time (ms)
Per-Subject Waveform from Global Top Attention Bin 0 (0–50 ms)
Amplitude
Time (ms)
Per-Subject Attention Weights — reach_vis_err
Attention Weight
Time (ms)

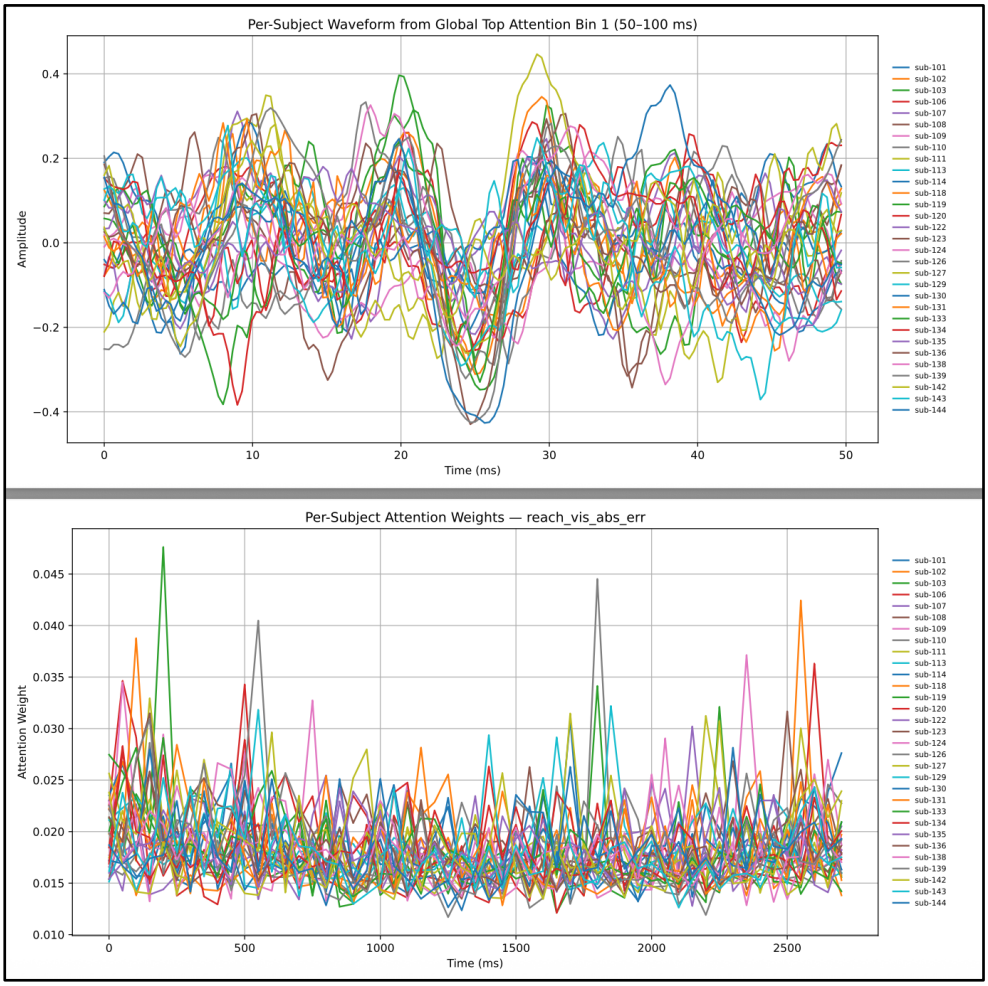
For the variable Reach_vis_abs_err



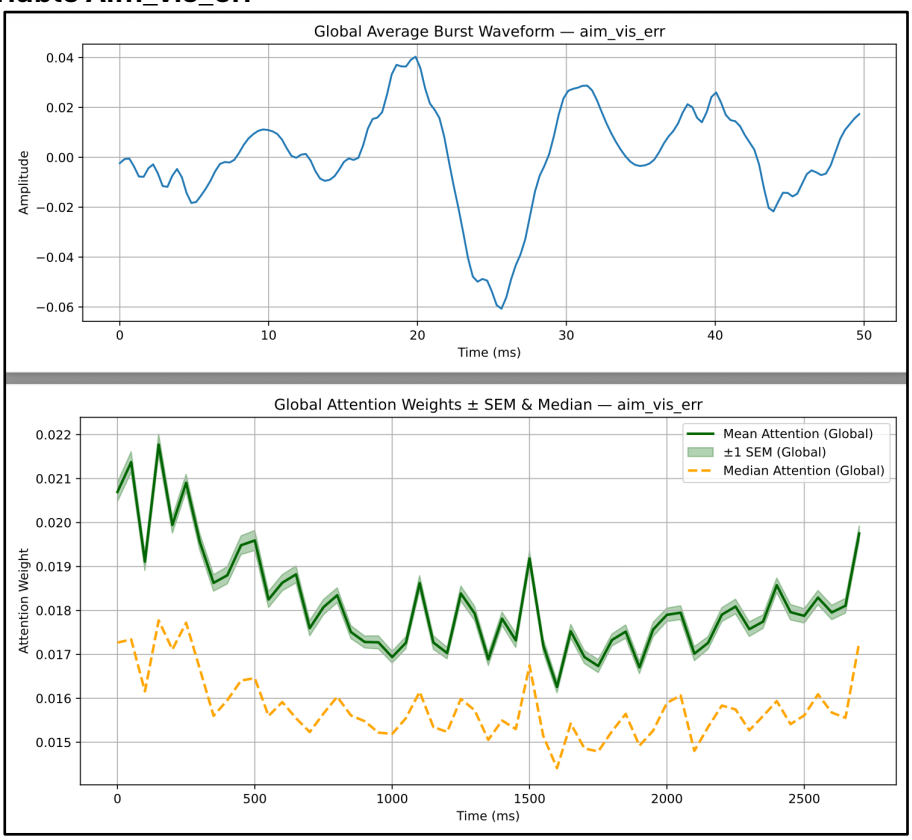


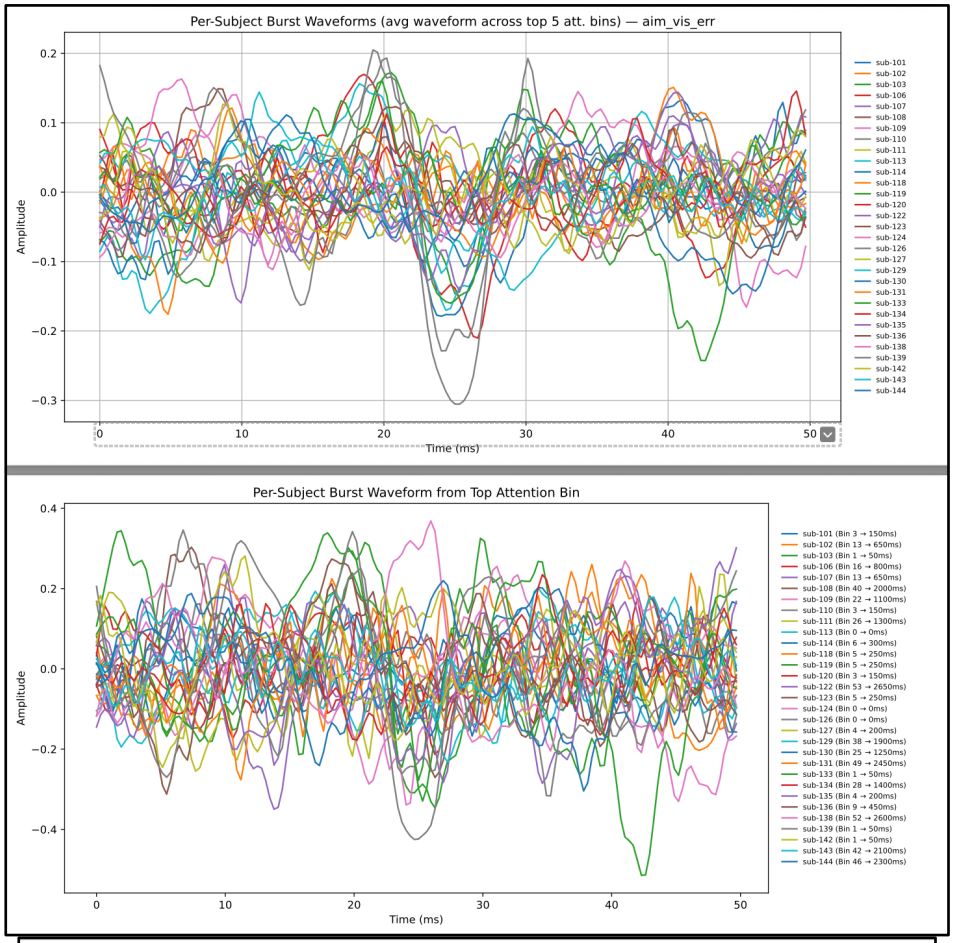


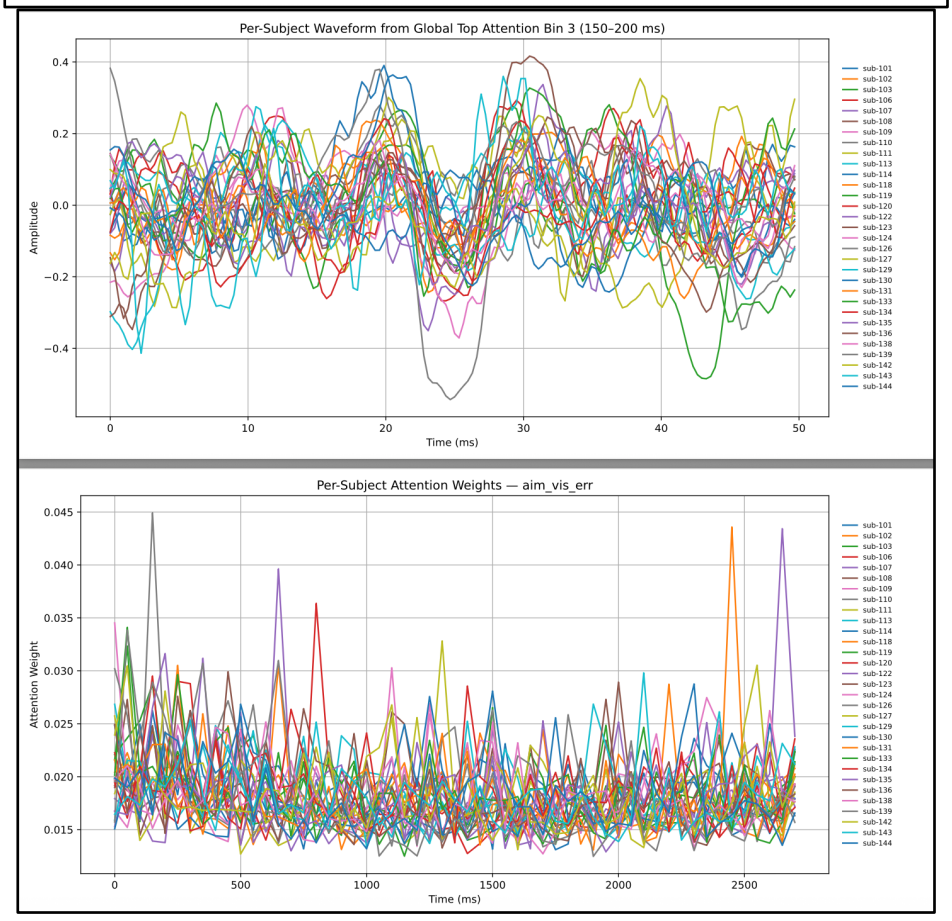




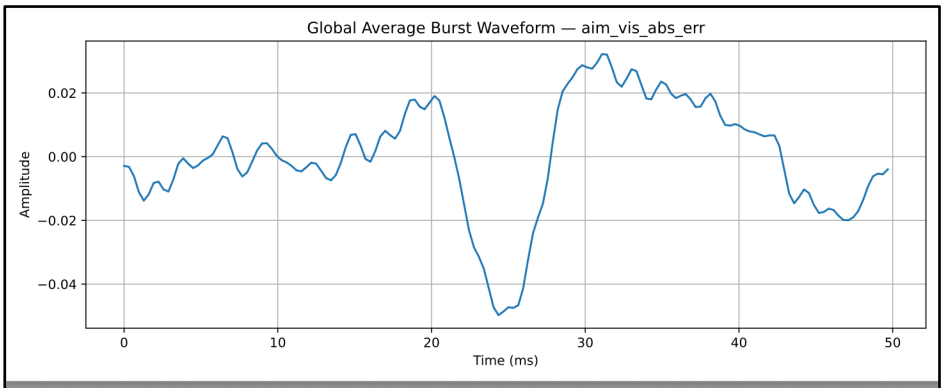
For the variable Aim_vis_err

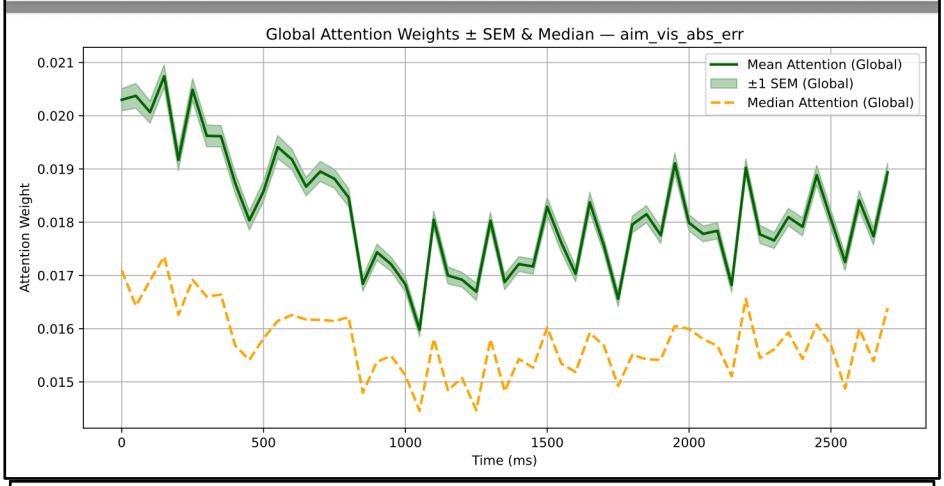


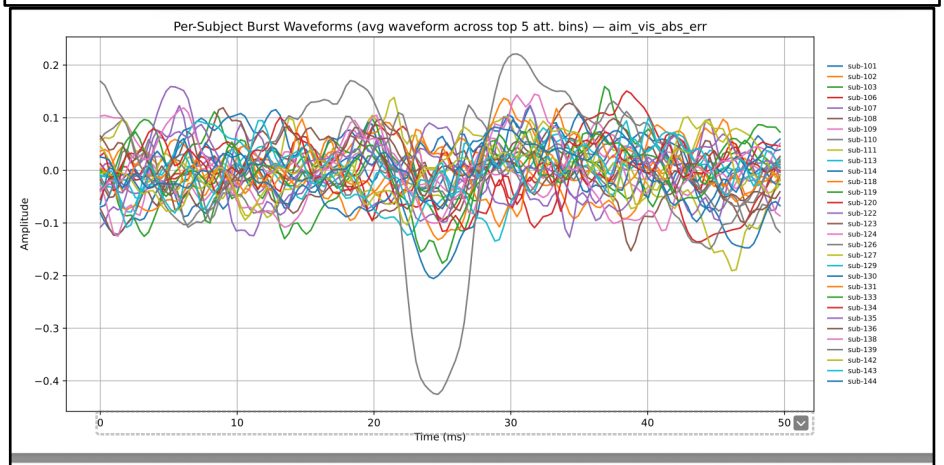

Per-Subject Burst Waveforms (avg waveform across top 5 att. bins) — aim_vis_err
Amplitude
Time (ms)
Per-Subject Burst Waveform from Top Attention Bin
Amplitude
Time (ms)

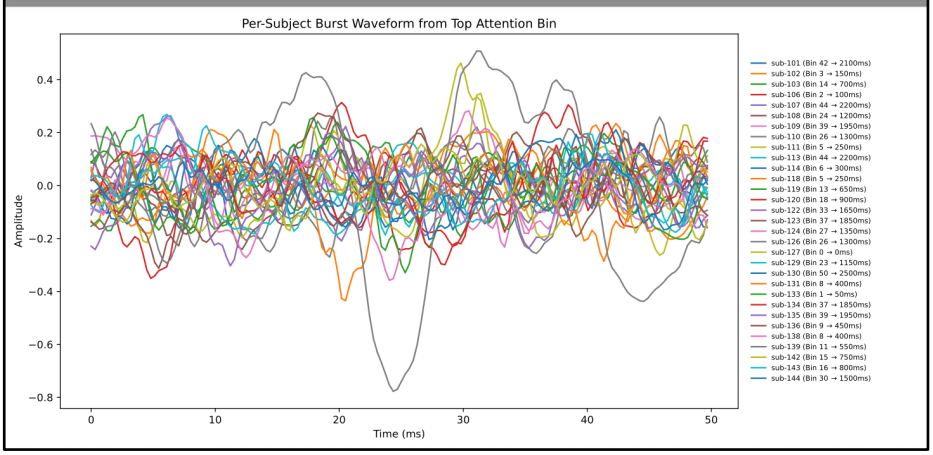

Per-Subject Waveform from Global Top Attention Bin 3 (150–200 ms)
Amplitude
Time (ms)
Per-Subject Attention Weights — aim_vis_err
Attention Weight
Time (ms)

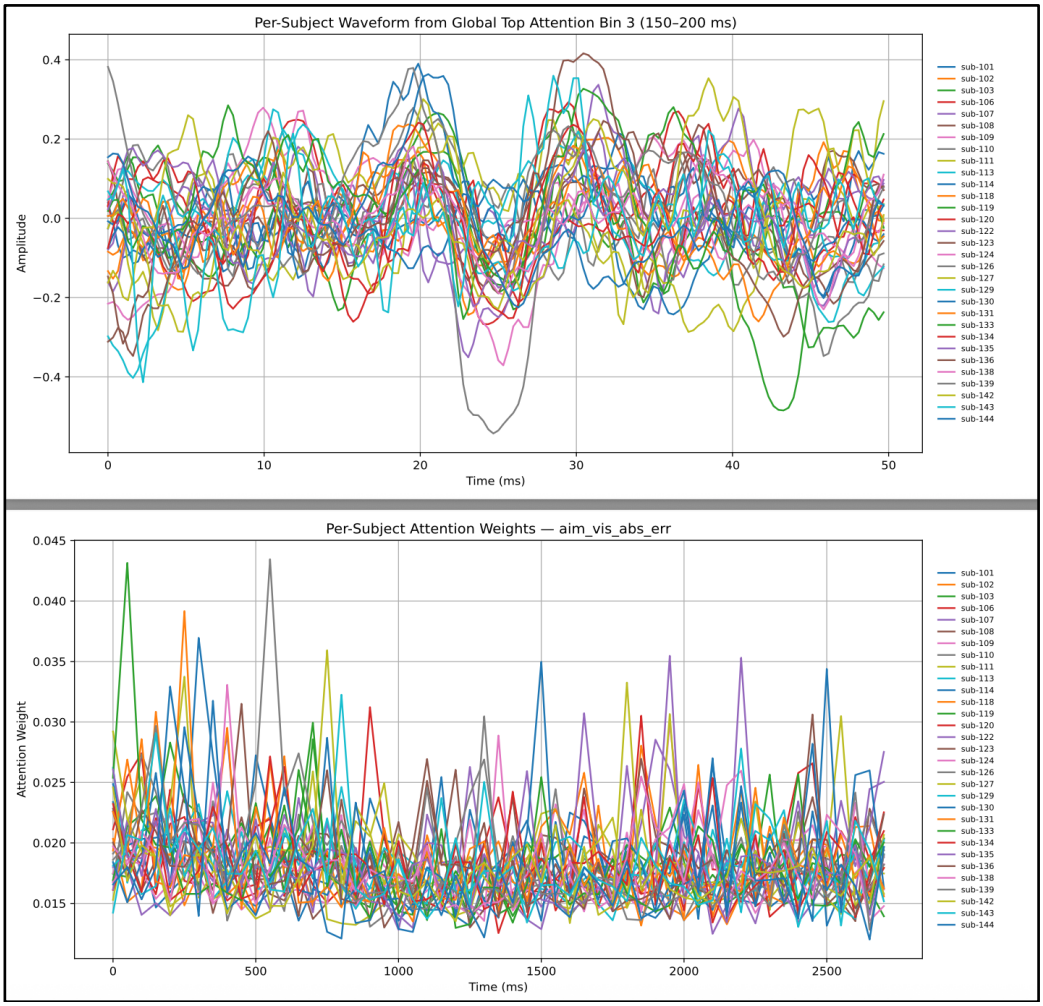
For the variable Aim_vis_abs_err











2. On 2 groups – Implicit and Explicit

In this analysis, we compared the Transformer regression model's performance across implicit and explicit conditions for predicting various behavioral metrics. For reach duration (reach_dur), the explicit condition outperformed the implicit one, with a higher R^2 score (0.41 vs. 0.26) and lower MSE, suggesting more accurate predictions when behavior is goal-directed. Similarly, for visual error metrics—both reach_vis_err and reach_vis_abs_err—the explicit condition consistently showed better predictive performance, indicating that conscious control enhances the link between burst features and visuomotor accuracy. However, for reaction time (reach_rt), the implicit condition yielded a higher R^2 score (0.43 vs. 0.29), suggesting that automatic processes may be more predictive of rapid responses. For aiming errors (both signed and absolute), the performance was comparable across both conditions, with a slight advantage to the implicit setting in some cases.

Overall, we conclude that explicit bursts tend to be more informative for predicting goal-directed motor behavior and visuomotor accuracy, while implicit bursts are more tightly linked to faster, automatic responses such as reaction time. This highlights a functional dissociation in how different behavioral aspects relate to burst dynamics under varying cognitive states.

a. **Plot Descriptions – for Implicit and Explicit**
    i. Attention Weights Across Time Bins (All Test Samples)
        1. Visualizes how much attention the model assigns to each time bin across all test samples.
        2. Shown as a heatmap: x-axis = time bins (0 to 54), y-axis = test samples.
        3. Helps identify which time periods are generally most informative for prediction.
    ii. Top Attention Bins – Waveform Shapes
        1. From the attention heatmap, the top 5 most attended bins are selected.
        2. For each of these bins, the average burst waveform shape is plotted.
        3. x-axis = burst feature index (1 to 156), y-axis = amplitude.
        4. Gives insight into how burst shapes vary in the most informative time bins.
    iii. Average Burst Waveform from Top Bins
        1. Combines all top 5 attention bins and averages their waveforms across all samples.
        2. Time-aligned waveform plotted with actual time in milliseconds.
        3. Shows the overall shape of neural bursts that contribute most to behavior prediction.
    iv. Mean Attention Weights ± SEM
        1. Plots the mean attention weight across all test samples, with shaded error bars (±1 SEM).
        2. Helps understand how confidently the model attends to each time bin.
        3. x-axis = time in ms; useful for identifying the most consistent attention windows.
    v. Frequency Spectrum (FFT) of Top Attention Bins
        1. For each top attention bin, computes the FFT (frequency spectrum) of the average waveform.
        2. x-axis = frequency (Hz), y-axis = magnitude.
        3. Shows the dominant frequency content (e.g., low vs high frequency bursts) in the most attended bins.
        4. Restricted to 0–50 Hz range for interpretability.

The results for the Implicit –
https://github.com/danclab/burst_transformer/blob/main/implicit.pdf

The results for the Explicit –
https://github.com/danclab/burst_transformer/blob/main/explicit.pdf