# Hypothesis-driven dimension reduction & source separation for time-domain EEG data

Hause Lin, Mike X Cohen

Slides & code: git.io/JUPXf

## Outline



What are and why use multivariate methods?

Generalized eigendecomposition (GED)

Using simulations to understand and implement GED

Empirical results

Slides & code: git.io/JUPXf

# What & why?

Cohen (2017, J Neuro Methods)
Cohen (2017, Trends in Neurosciences)
Grootswager et al. (2017, J Cog Neuro)

Slides & code: git.io/JUPXf

**Multivariate methods leverage EEG data's rich spatiotemporal structure**

ERPs highlight event-related activity and ↑ signal-to-noise ratio
- *information is lost during averaging*

Univariate methods (ERPs) analyze individual electrodes separately
- *neurocognitive processes are encoded in spatiotemporal patterns*

Volume conduction: neural sources at mixed at the electrodes
- *ERPs don't unmix overlapping spatial and temporal information*

Multivariate methods address these problems!

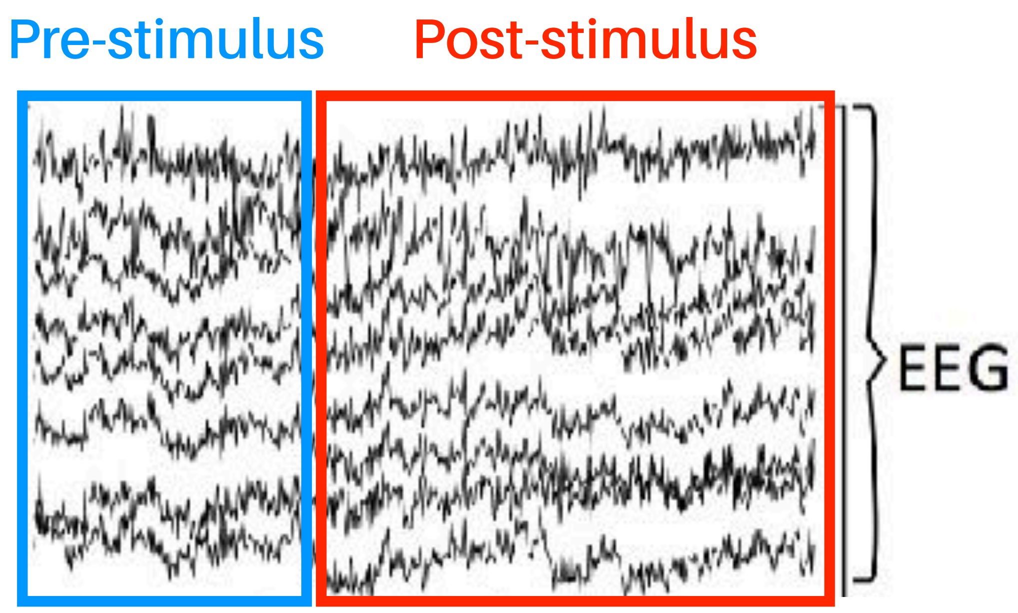
## Generalized eigendecomposition (GED) framework

Cohen (2017, eLife)
Cohen (2017, J Neuro Methods)
Parra & Sajda (2003, J Mach Learn Res)
Parra et al. (2005, NeuroImage)

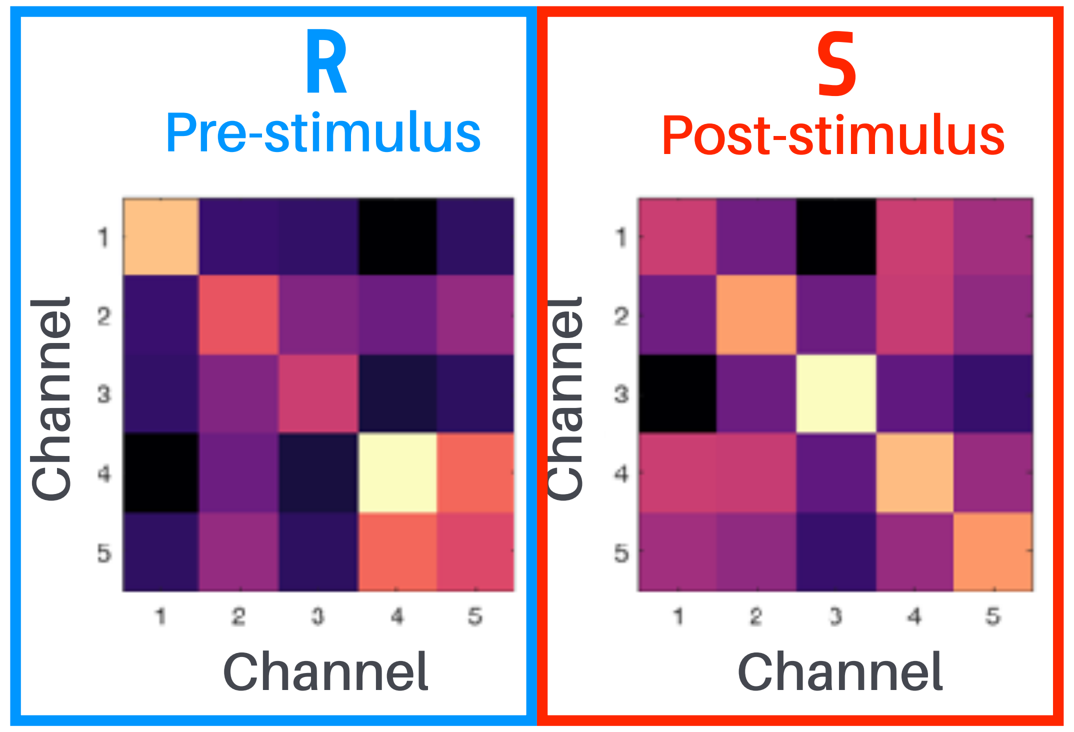
Slides & code: git.io/JUPXf

# Ideal (and designed) for testing hypotheses

Directly contrasts two “conditions”
- hypothesis-driven (not “blind” source separation)
- represented by covariance matrices **R** and **S** (**r**eference, **s**ignal)




Raw EEG data (1 trial)




Compute channel-by-channel **covariance matrices** for each trial

**Multivariate:** Considers relationships/ between channels

```
% dat: trial 1 chan-by-time matrix
dat = EEG.data(:,:,1);
% remove third dimension
dat = squeeze(dat);
```

```
% datR: select first 100 timepoints (pre-stimulus)
datR = dat(:,1:100);
R = cov(datR');
% datS: select remainder timepoints (post-stimulus)
datS = dat(:,101:end);
S = cov(datS');
```

# Compute R and S covariance matrices separately for each trial

```
% MATLAB implementation (assumes EEG variable is EEGLAB's EEG structure)

% STEP 1: initialize empty covariance matrices (chan-by-chan covariance matrix)
[R, S] = deal(zeros(EEG.nbchan));  % R (reference), S (signal)

% STEP 2: compute covariance R, S matrices for each trial
for ti=1:EEG.trials  % for each trial
    datR = squeeze(EEG.data(:,1:100,ti));  % get pre-stimulus data
    R = R + cov(datR');  % compute covariance matrix R and sum them

    datS = squeeze(EEG.data(:,101:end,ti));  % get post-stimulus data
    S = S + cov(datS');  % compute covariance matrix S and sum them
end

% STEP 3: compute average covariance (divide by total number of trials)
R = R./EEG.trials;
S = S./EEG.trials;
```

## Generalized eigendecomposition (GED) framework

Cohen (2017, eLife)
Cohen (2017, J Neuro Methods)
Parra & Sajda (2003, J Mach Learn Res)
Parra et al. (2005, NeuroImage)

Slides & code: git.io/JUPXf

## Conceptually simple and computationally efficient

Similar to principal component analysis (PCA)
- PCA: simplified, special case of GED
  - solves the basic eigenvalue equation: $\mathbf{S}w = w\lambda$
  - `[evecs,evals] = eig(S);`
- GED generalizes equation to two matrices: $\mathbf{SW} = \mathbf{RW\Lambda}$
  - `[evecs,evals] = eig(S,R);`

## Generalized eigendecomposition (GED) framework

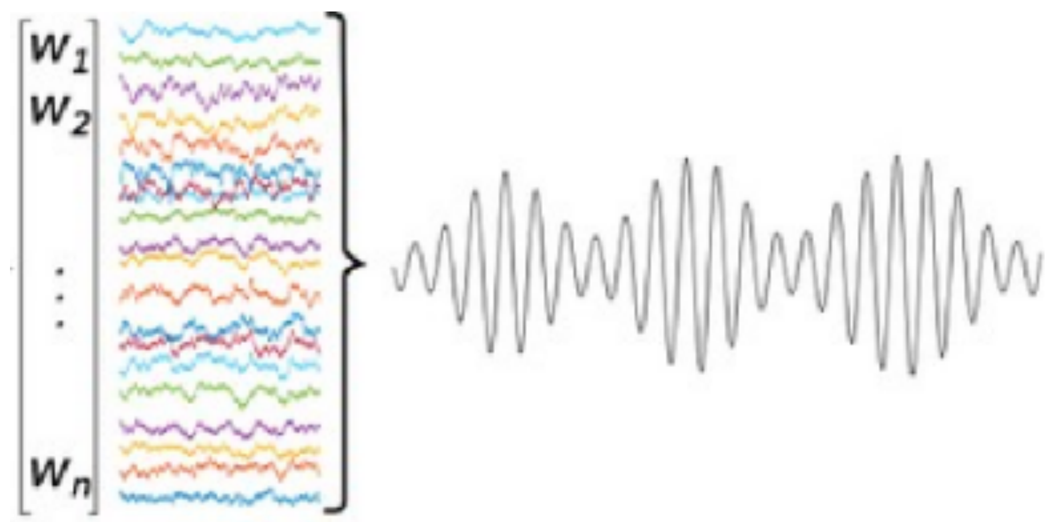
Cohen (2017, eLife)
Cohen (2017, J Neuro Methods)
Parra & Sajda (2003, J Mach Learn Res)
Parra et al. (2005, NeuroImage)

Slides & code: git.io/JUPXf

## Reduces data dimensions and separates sources

Eigenvectors (`evecs`): spatial filters for computing components
Eigenvalues (`evals`): importance of components

Eigenvectors/components are **independent** but **not orthogonal**
- PCA doesn't separate sources well because sources are orthogonal




Apply weights/filters to original data to unmix sources

```
% topography for sources/components 1 & 2
topography_component1 = evecs(:,1)*S;
topography_component2 = evecs(:,2)*S;

% time series for sources/components 1 & 2
comp_ts = evecs(:,1:2)'*reshape(EEG.data,EEG.nbchan,[]);
comp_ts = reshape(comp_ts,[2 EEG.pnts EEG.trials]);
```

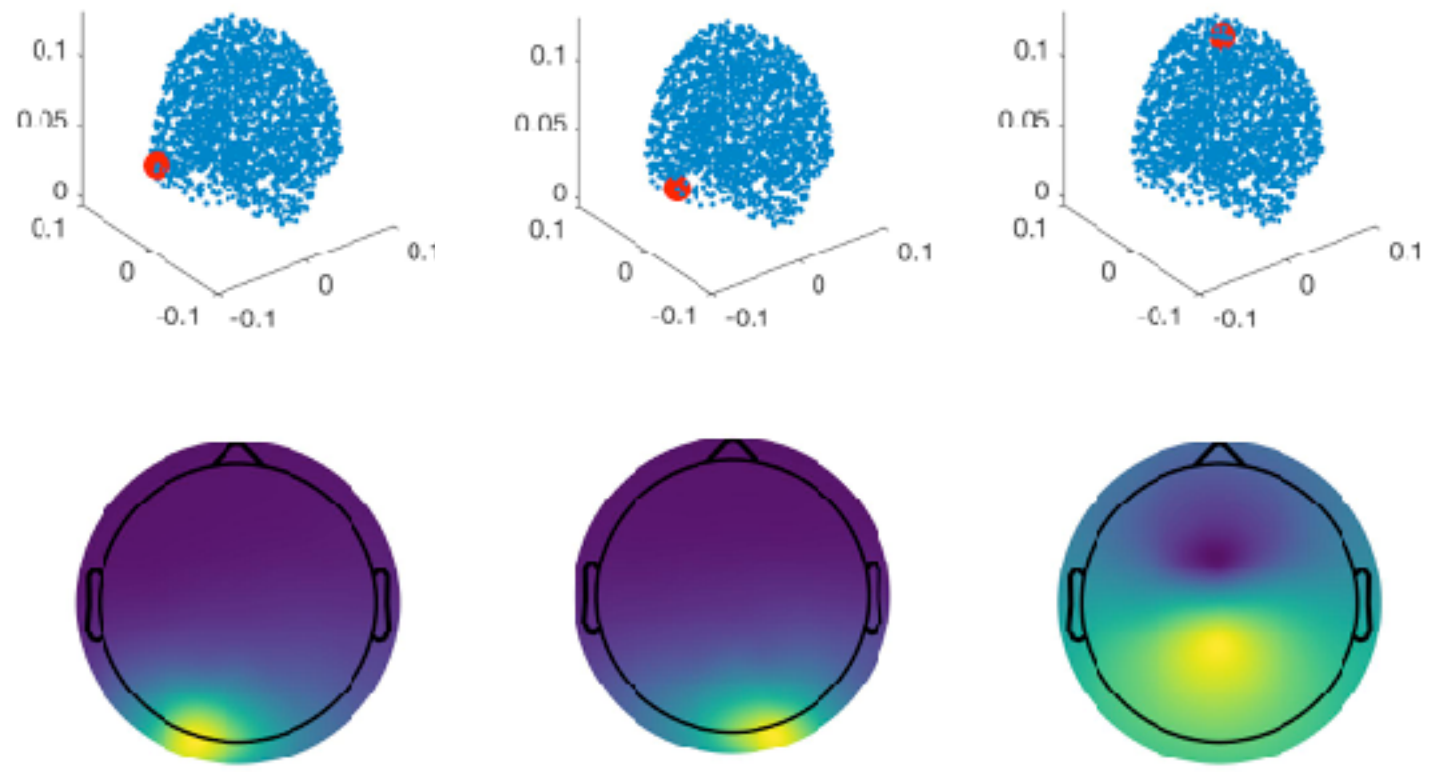
8 Simulation

# Defining simulation parameters

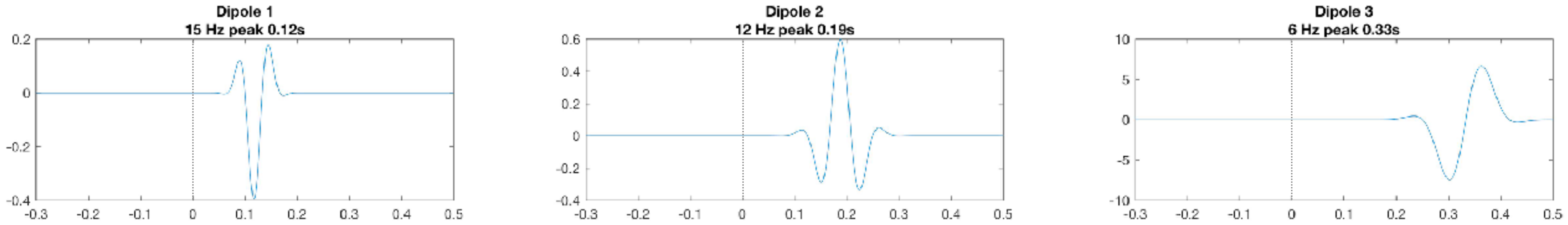
Lin & Cohen (in prep)

Slides & code: git.io/JUPXf

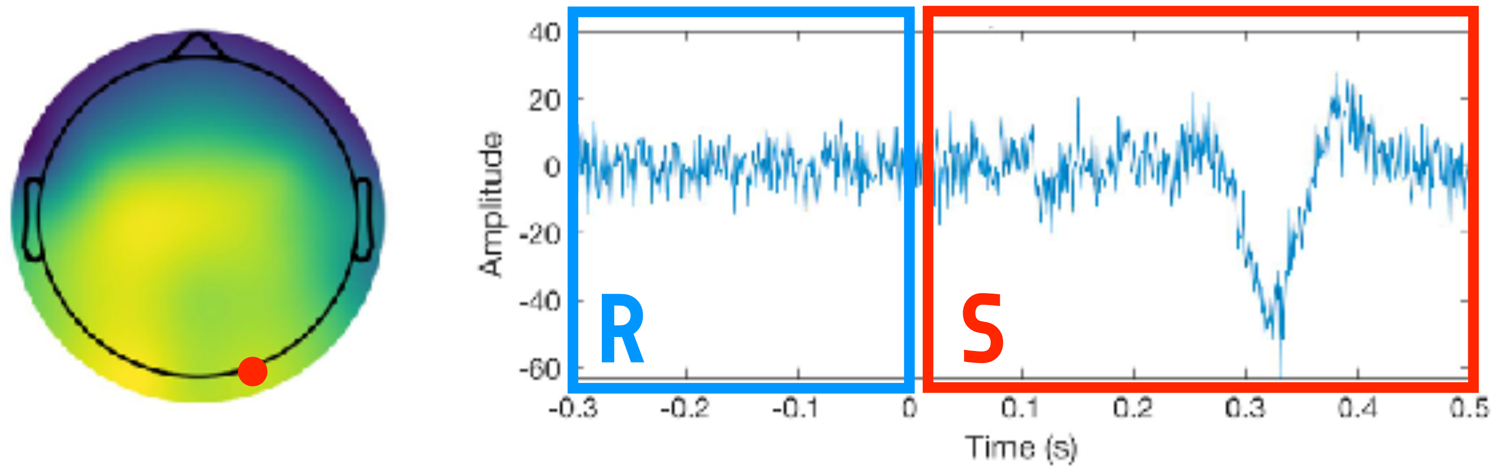
## Spatial locations and topographies of 3 dipole sources



## Simulated pure neural signals




## Mixing of neural signals and noise + projection to the scalp

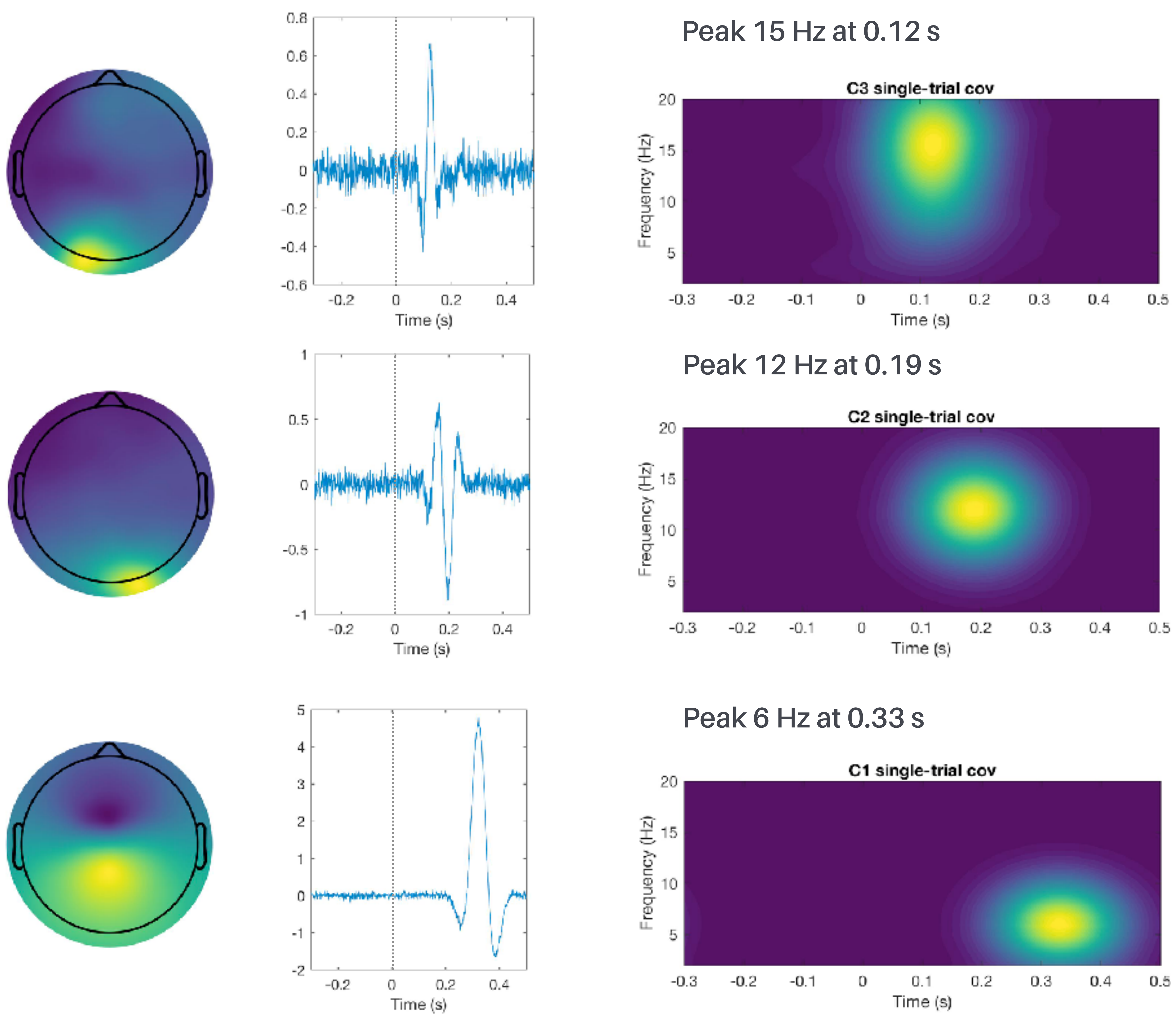



"Observed" topography and ERP

# Recovering parameters from noisy simulated data

Lin & Cohen (in prep)

## GED recovers simulated spatiotemporal and spectral dynamics



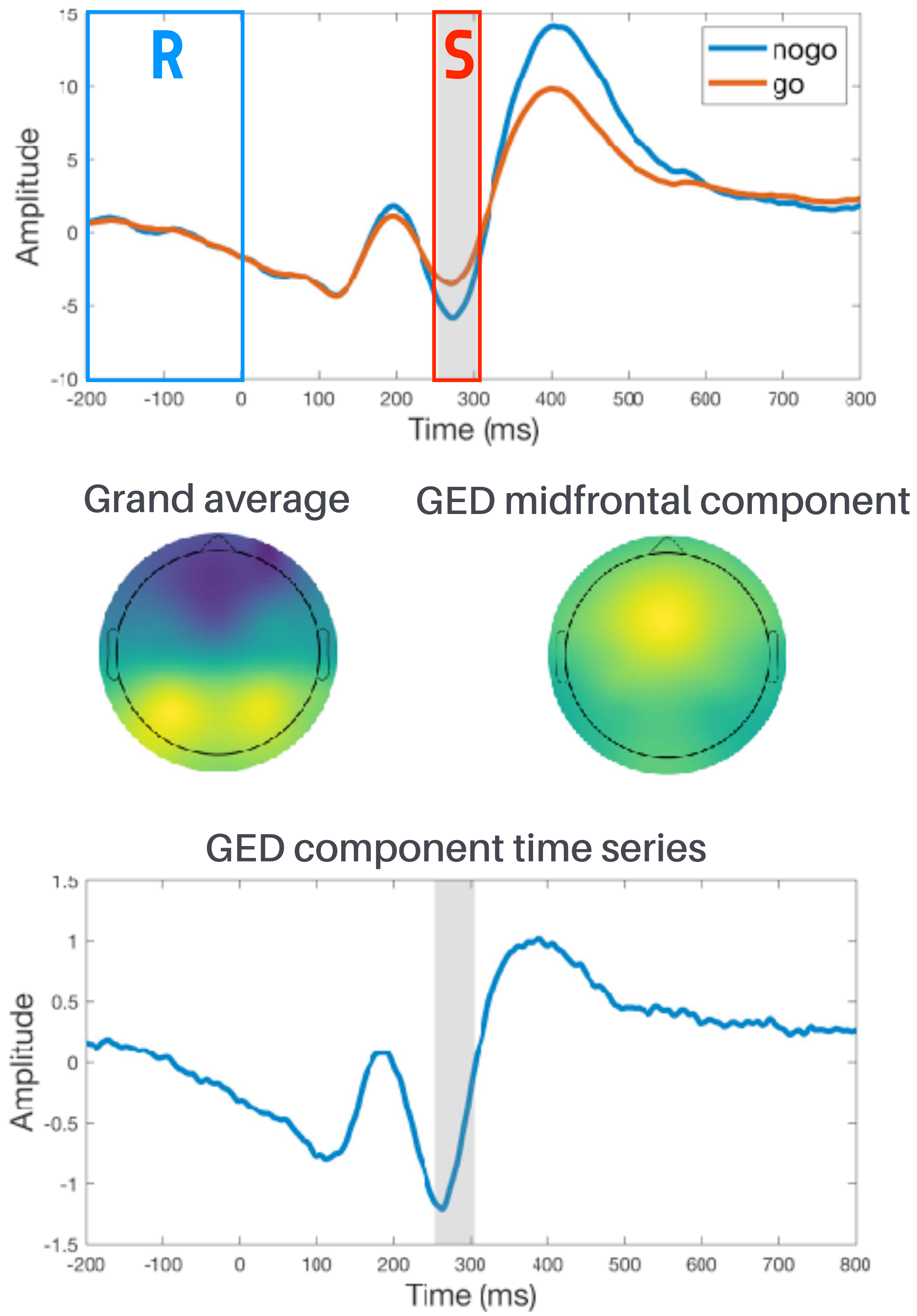

Slides & code: git.io/JUPXf

10 GED on real data

# Conflict-monitoring

Lin & Cohen (in prep)

## Go/no-go task N2 ERP (254 to 304 ms)




**GED parameters**

**R** covariance matrix

- computed using <u>all trials</u>, -200 to 0 ms pre-stimulus

**S** covariance matrix

- computed using <u>all trials</u>, 254 to 304 ms post-stimulus

Slides & code: git.io/JUPXf

# Feedback processing

Lin & Cohen (in prep)

## Reward positivity ERP at 235 to 285 ms


**GED parameters**

R covariance matrix
- computed using <u>all trials</u>, -200 to 0 ms pre-feedback

S covariance matrix
- computed using <u>all trials</u>, 235 to 285 ms post-feedback

**Generalized eigendecomposition: Flexible multivariate method**

Ideal for experimental research (hypothesis testing)

Reduces dimensionality and separates sources (*hypothesis-driven* source-separation)

Components/sources are independent but non-orthogonal (aligns with actual brain dynamics)