DONDERS INSTITUTE

Radboud University

Non-parametric statistical testing with clusters

Tzvetan Popov

Central Institute of Mental Health, Mannheim, Germany

Talk outline

Inferential statistics

Channel-level statistics

- parametric
- non-parametric
- clustering

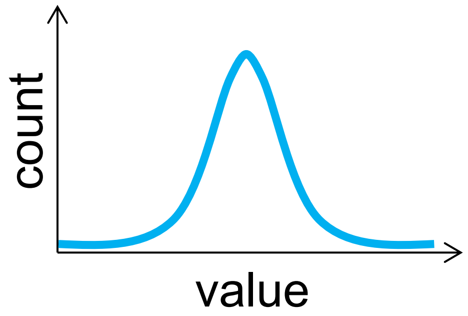
Source-level statistics

Inferential parametric statistics

You make N observation and want to find whether some hypothesis H1 is true

Step 1: Gathering data

Observation	Value
0	2.5
1	-3.2
.	
.	
.	
N	2.4

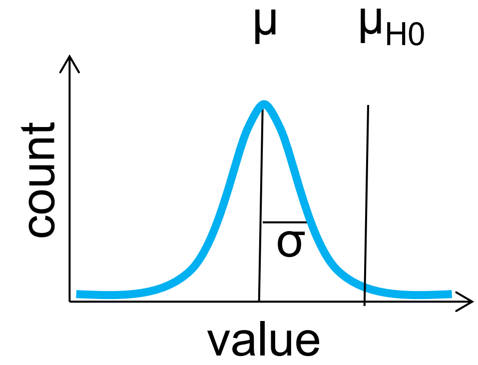


Inferential parametric statistics

You make N observation and want to find whether some hypothesis H1 is true

Step 2: Statistical testing

Observation	Value
0	2.5
1	-3.2
.	
.	
.	
N	2.4



Determine probability of t under H0

$$t = \frac{\mu - \mu_{H0}}{\sigma / \sqrt{N}}$$

If t sufficiently unlikely, reject H0

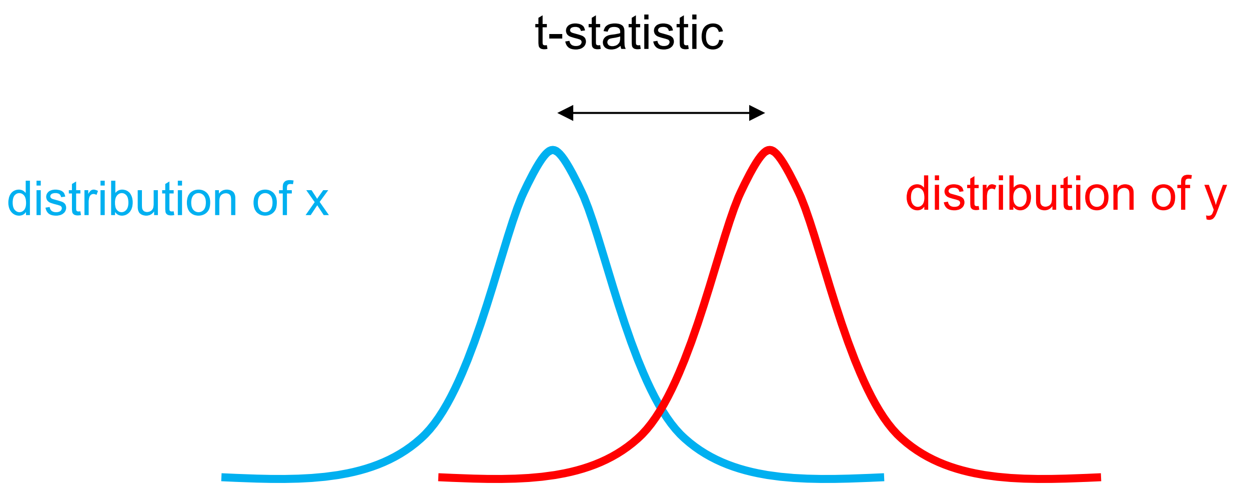
Inferential parametric statistics

Observations in condition 1:

{x1, x2, x3, x4, …}

Observations in condition 2:

{y1, y2, y3, y4, …}

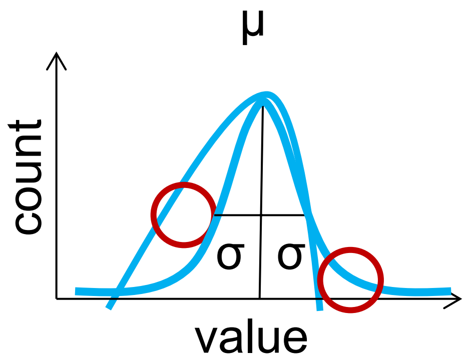


Parametric statistical testing

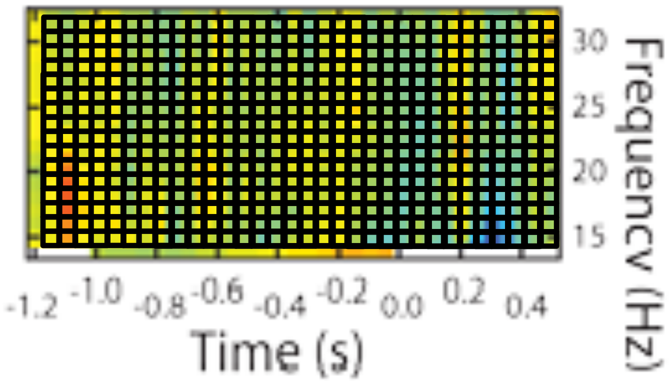
You make N observation and want to find whether some hypothesis H1 is true.

The first problem is that this requires a *known distribution* of the test statistic.

Observation	Value
0	2.5
1	-3.2
.	
.	
.	
N	2.4



The second problem is that of multiple comparisons



16 *30 time-frequency tiles, i.e. 480 comparisons.

t-test with α = 0.05 (chance of false alarm rate) **for one test**

Chance of one false alarm in 480 tests: 99.99...%

Or: 480*0.05 = 24 false alarms expected

The multiple comparison problem

Whole-brain analysis

306 channels

100 timepoints

50 frequencies

- 1.530.000 statistical tests

5% chance of false alarm in every test

- 76.500 false alarms

Solutions to control the FWER

- Bonferroni correction
- Use the false discovery rate
- Use a Monte Carlo approximation of the randomization distribution of the maximum statistic

```
cfg  = [];
cfg.method   = ‘analytic’
cfg.correctm = ‘bonferroni’
ERPstats     = ft_timelockstatistics(cfg, ERP);
```

```
cfg  = [];
cfg.method   = ‘analytic’
cfg.correctm = ‘fdr’
ERPstats     = ft_timelockstatistics(cfg, ERP);
```

```
cfg  = [];
cfg.method   = ‘montecarlo’
cfg.correctm = ‘max’
TFRstats     = ft_freqstatistics(cfg, TFR);
```

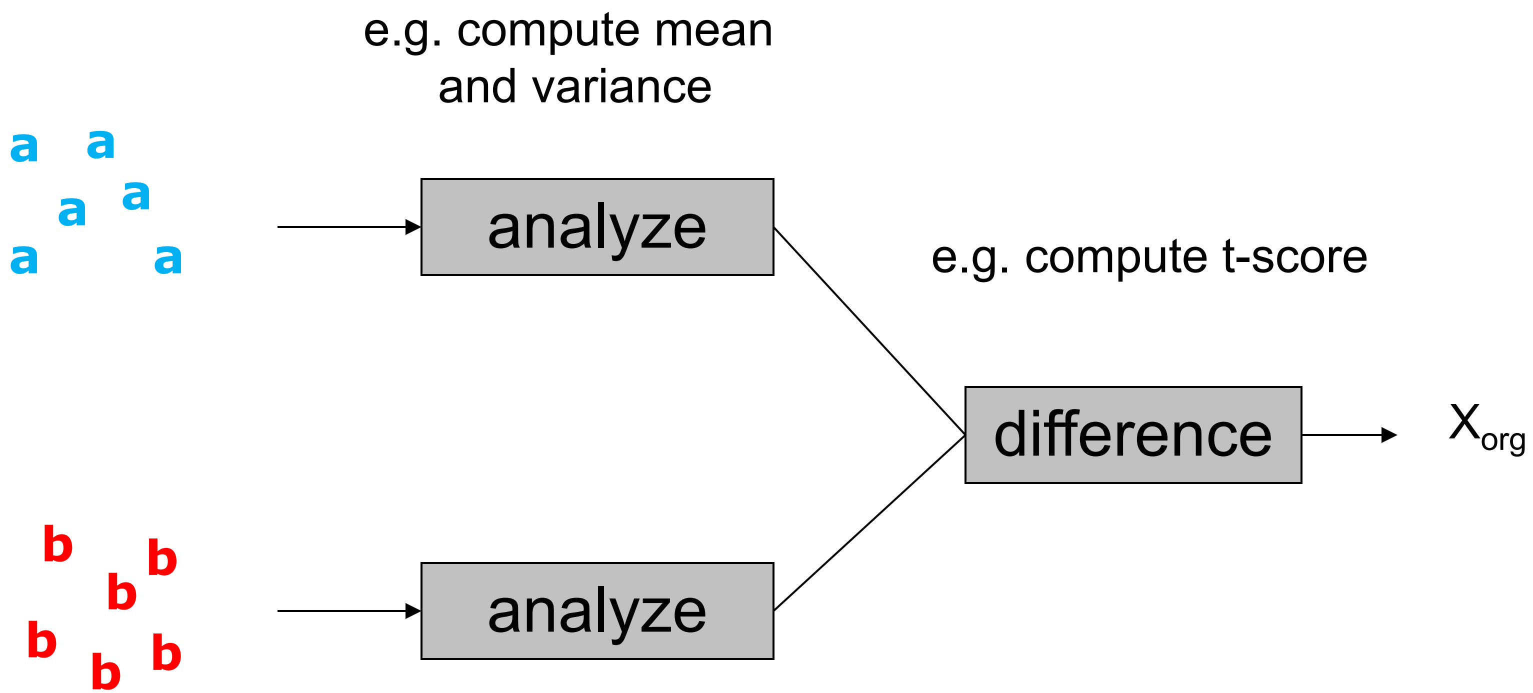
Randomization test: general principle

- *Independent variable: condition*
- *Dependent variable: data*

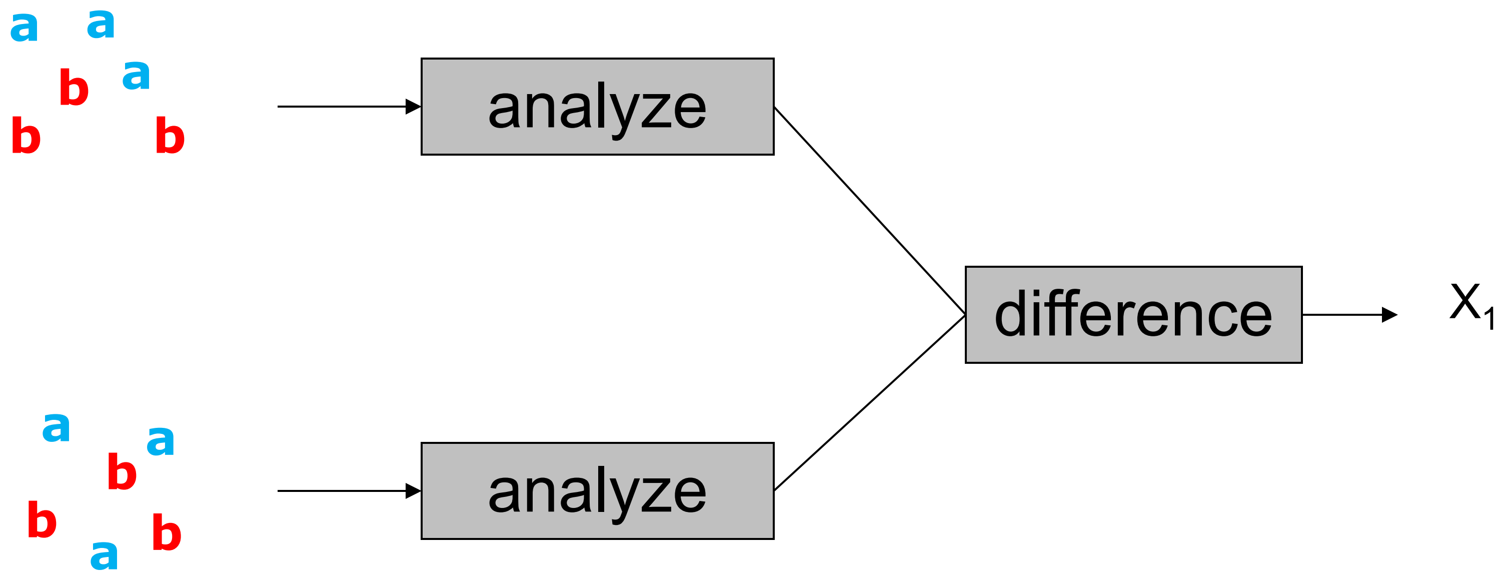
H0: the data is **independent** from the condition in which it was observed

The data in the two conditions is **not** different

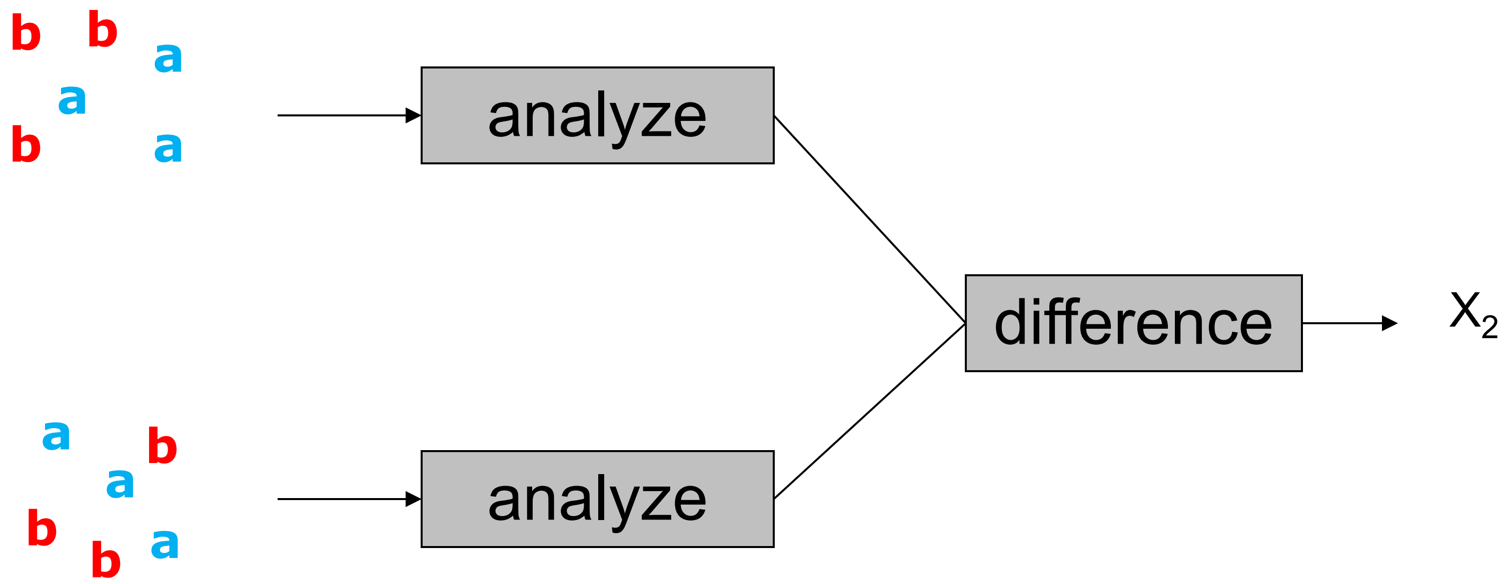
Randomization approach



Randomization approach



Randomization approach



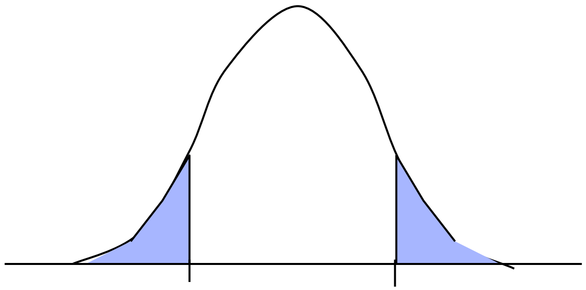
Randomization approach

b b a
a
b a

a b
a
b b a

X_2

Distribution of “x” can take any shape



Non-parametric statistics

Randomization of independent variable

Hypothesis is about data, not about the specific parameter

Randomization distribution of the statistic of interest “x” is approximated using Monte-Carlo approach

H0 is tested by comparing the observed statistic against the randomization distribution

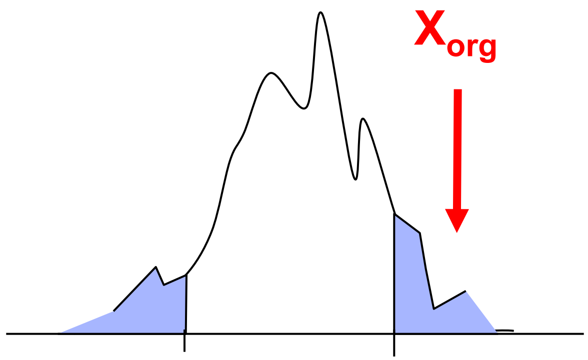
Avoid the multiple comparison problem

The statistic "x" can be anything

Rather than testing everything, only test the most extreme observation (i.e. the max statistic)

Compute the randomization distribution for the most extreme statistic

Note that often we compute **two** extrema, one for each tail



Increasing the sensitivity

Conventional is univariate parametric

Our approach is to consider the data

- Many channels, timepoints, frequencies
- Massive univariate
- Multiple comparison problem

There is quite some structure in the data

Increasing the sensitivity

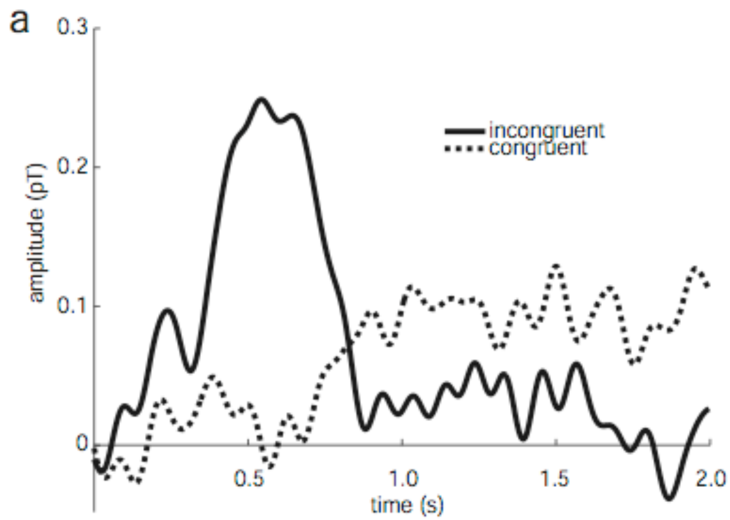
channel/time/frequency points are not independent

neighbouring channel/time/frequency points are expected to show similar behaviour

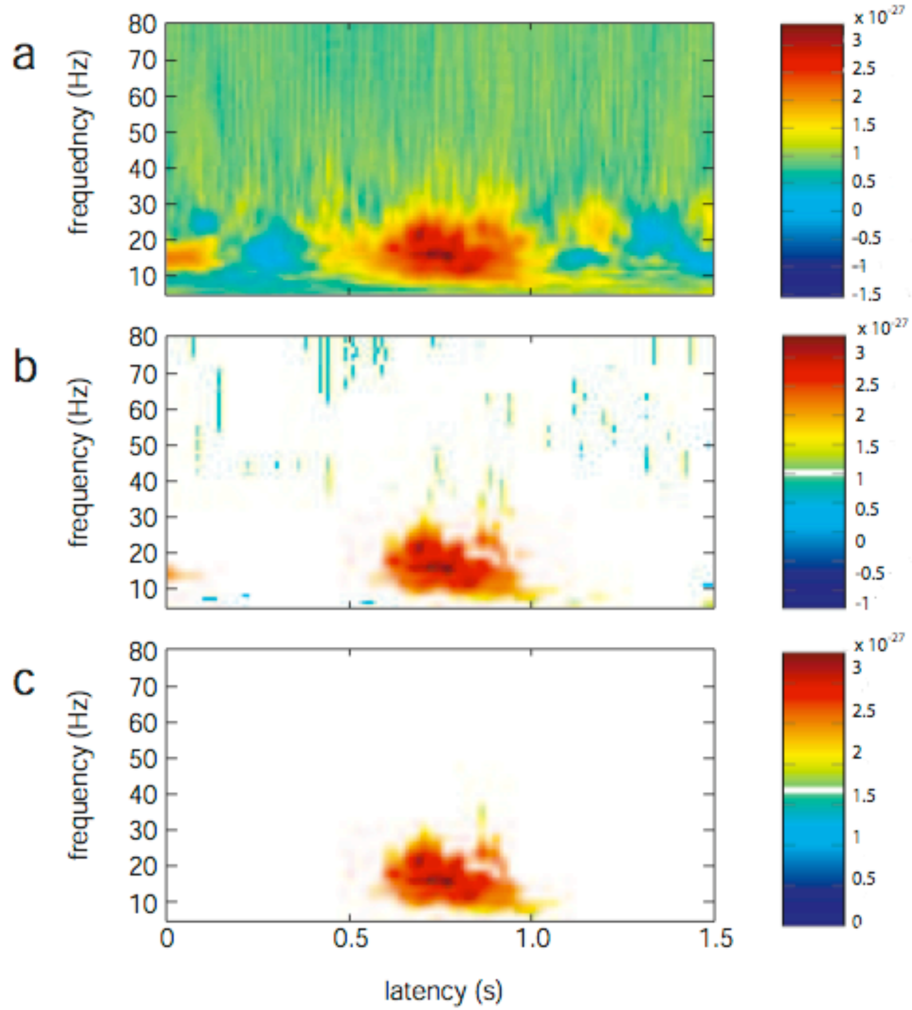
combine neighbouring samples into clusters -> "accumulate the evidence" = cluster-based statistics

avoid the MCP by comparing the largest observed cluster versus the randomization distribution of the largest clusters

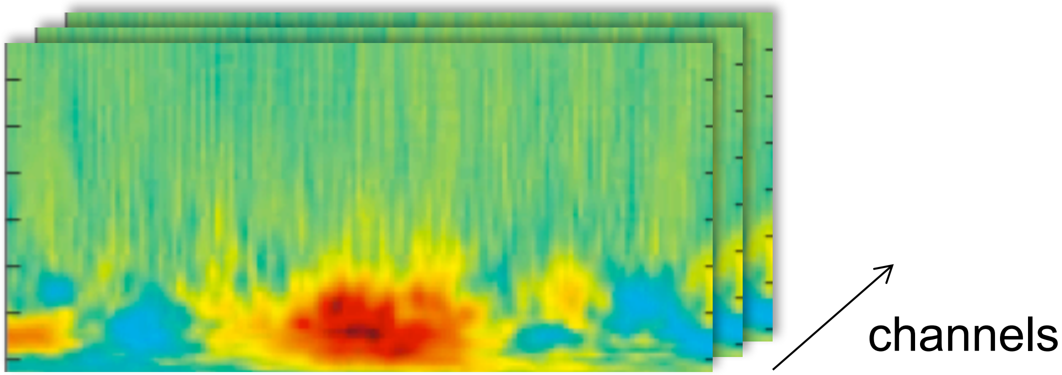
Clustering in time

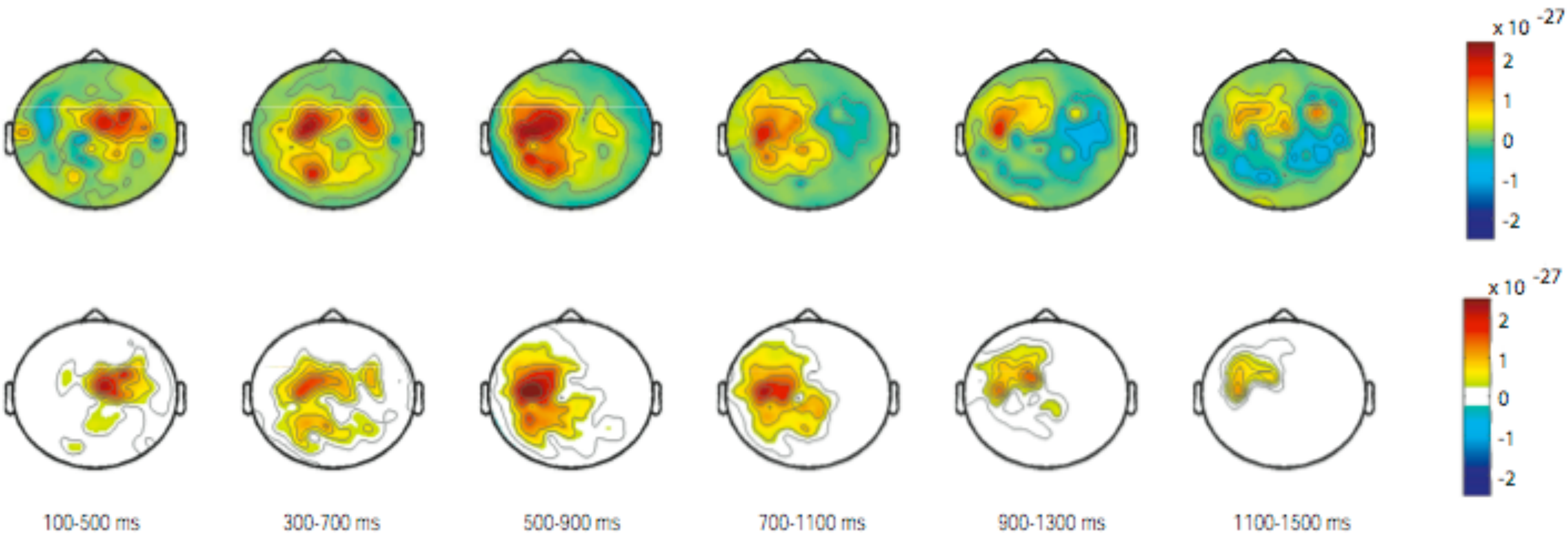


Clustering in time and frequency



Clustering in time, frequency and space





http://www.fieldtriptoolbox.org

Toy example

Toy example: Original observation

null hypothesis: condition A = condition B

Condition A	Condition B
S1_a	S1_b
S2_a	S2_b
S3_a	S3_b
S4_a	S4_b
S5_a	S5_b
S6_a	S6_b
S7_a	S7_b
S8_a	S8_b
S9_a	S9_b
S10_a	S10_b

Toy example: 1st permutation

null hypothesis: condition A = condition B

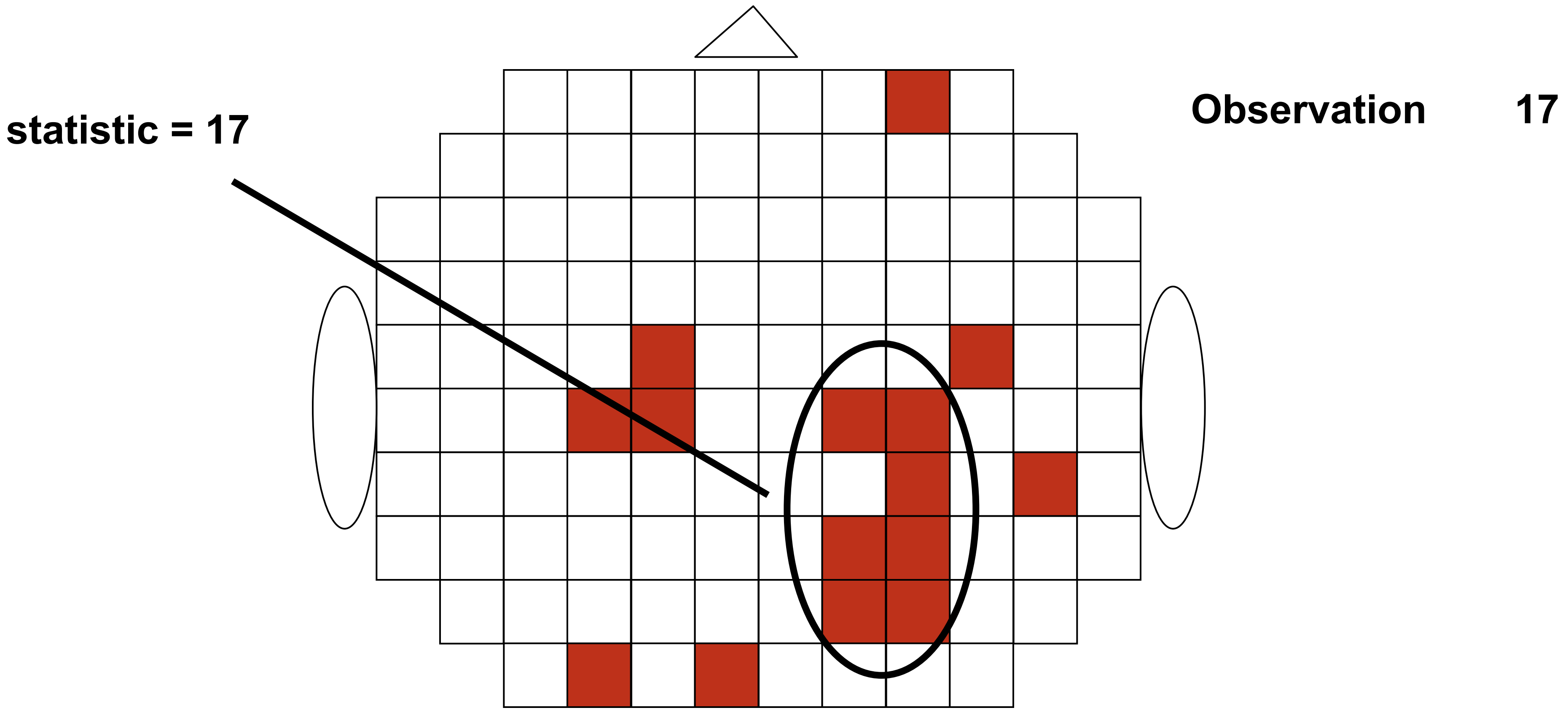
Condition A	Condition B
S1_a	S1_b
S2_b	S2_a
S3_a	S3_b
S4_a	S4_b
S5_b	S5_a
S6_b	S6_a
S7_a	S7_b
S8_a	S8_b
S9_a	S9_b
S10_b	S10_a

Toy example: 2nd permutation

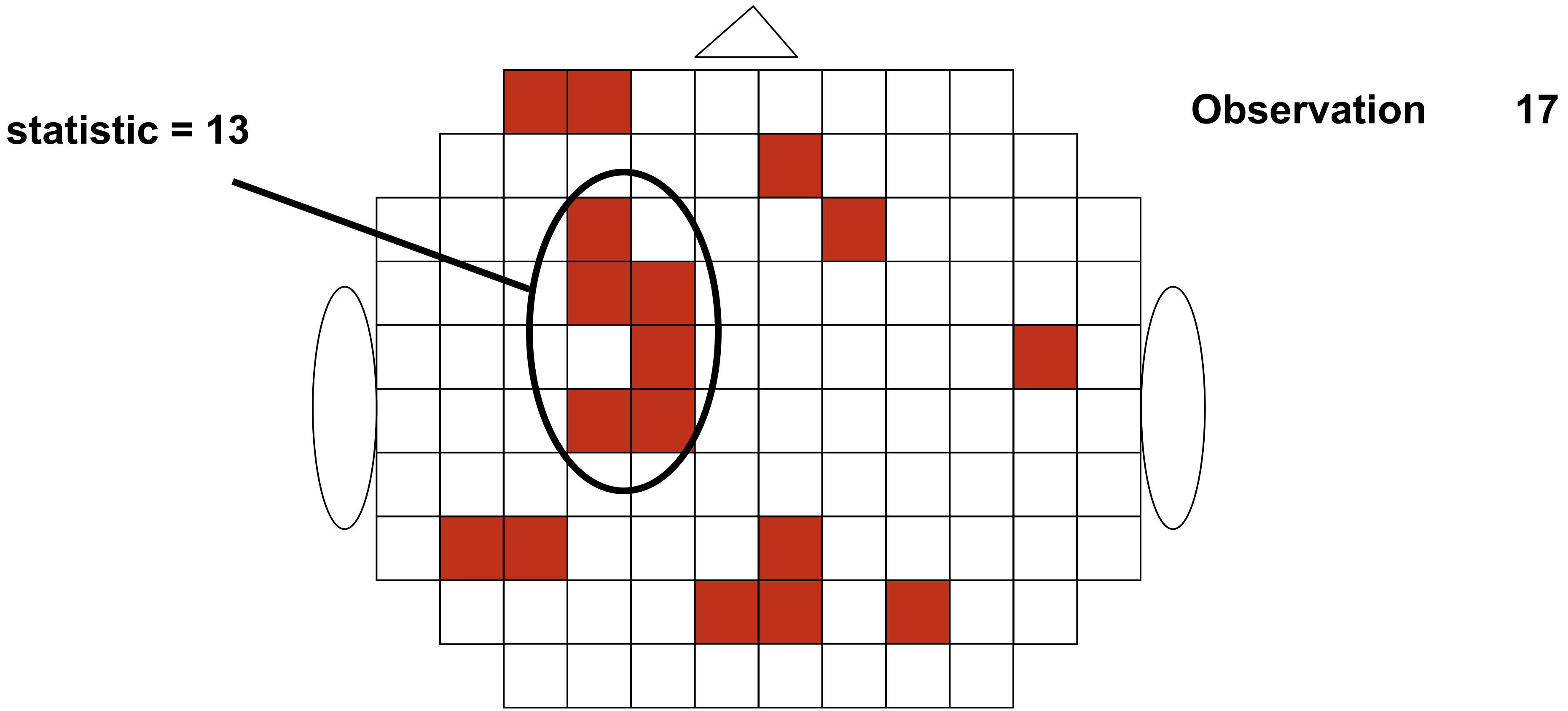
null hypothesis: condition A = condition B

Condition A	Condition B
S1_b	S1_a
S2_a	S2_b
S3_b	S3_a
S4_a	S4_b
S5_a	S5_b
S6_b	S6_a
S7_a	S7_b
S8_b	S8_a
S9_b	S9_a
S10_a	S10_b

Toy example: Original observation

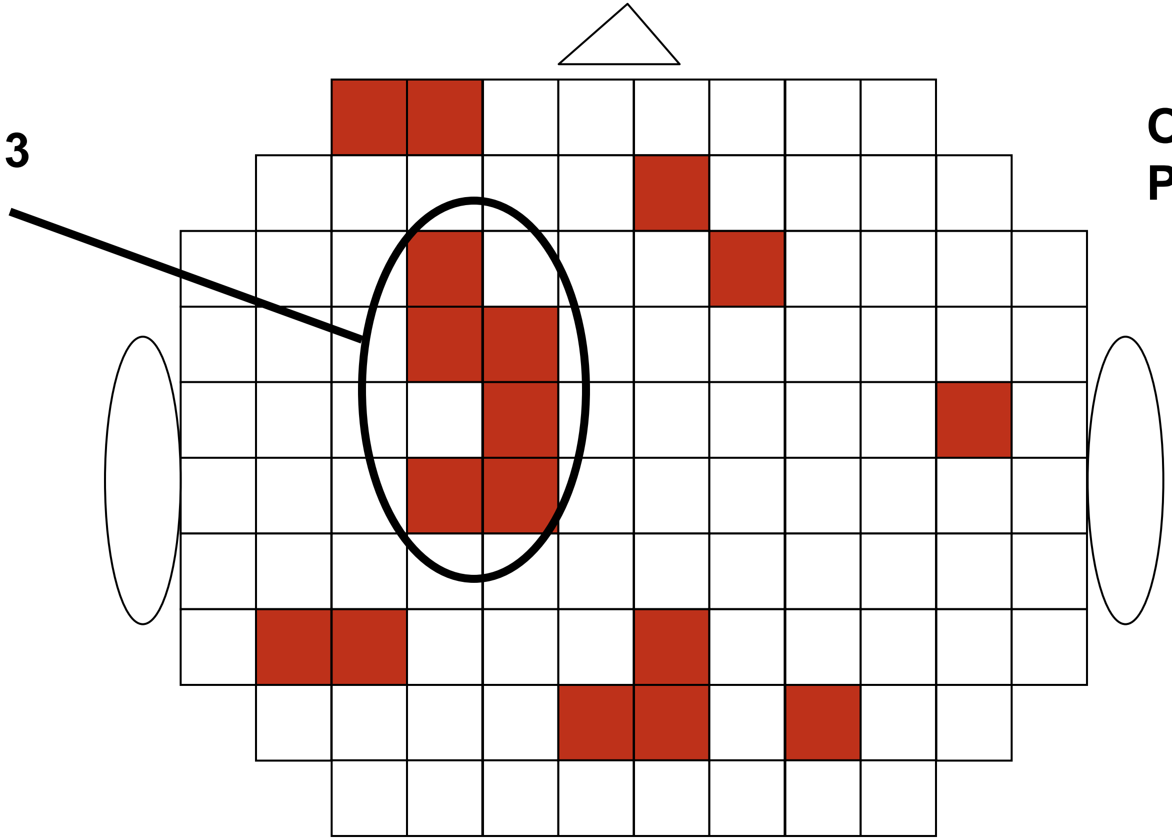


Toy example: 1st permutation



Toy example: 1st permutation

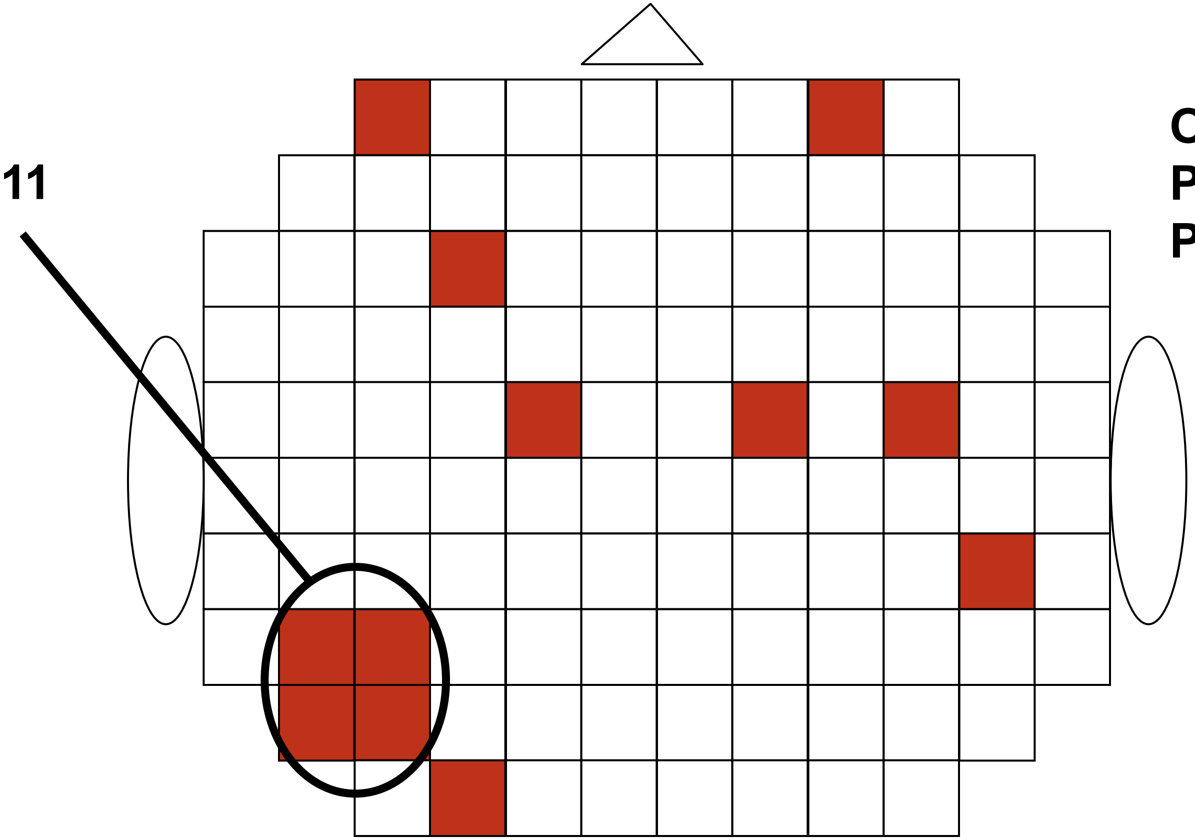
statistic = 13



Observation 17
Permutation 1 13

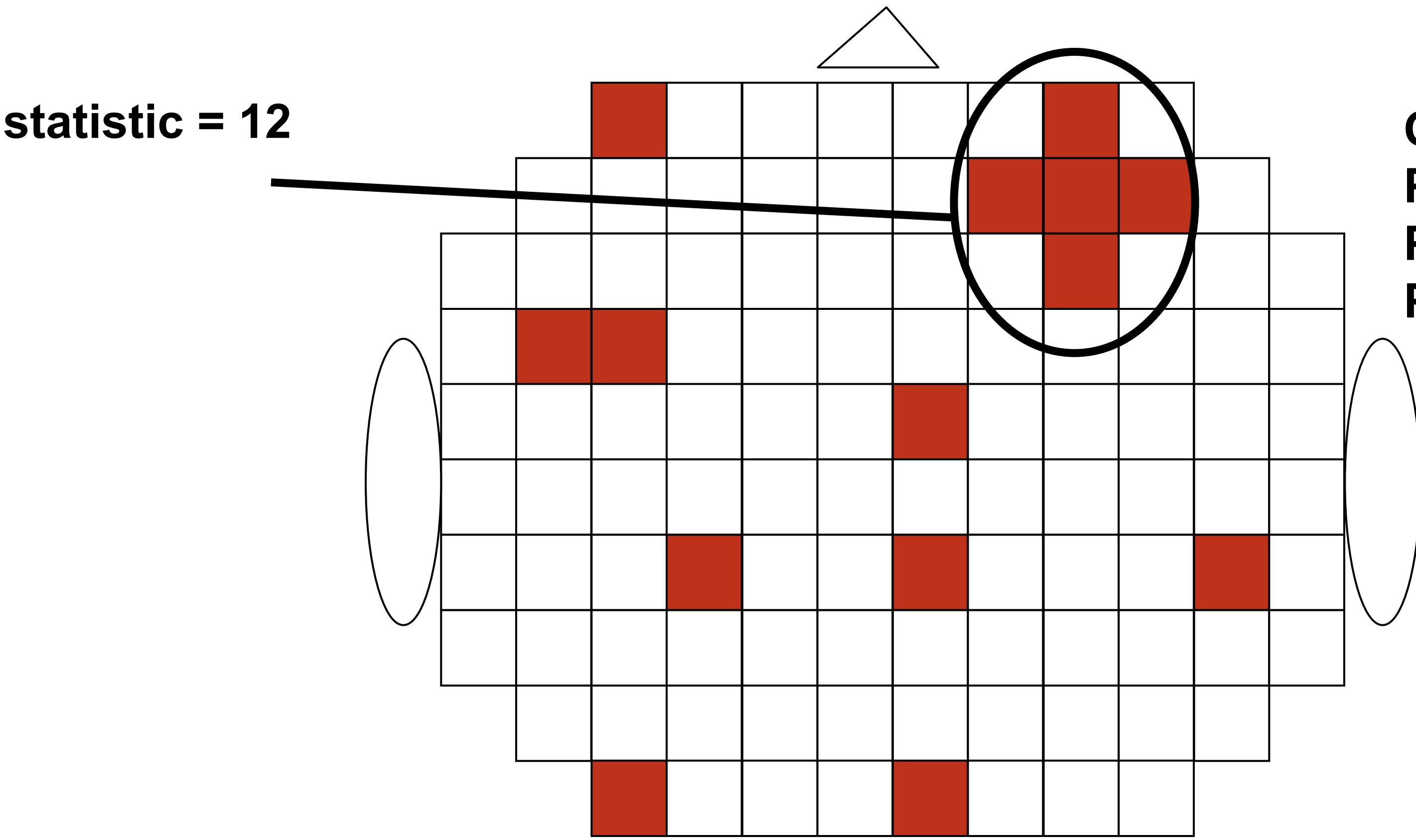
Toy example: 2nd permutation

statistic = 11



Observation	**17**
Permutation 1	**13**
Permutation 2	**11**

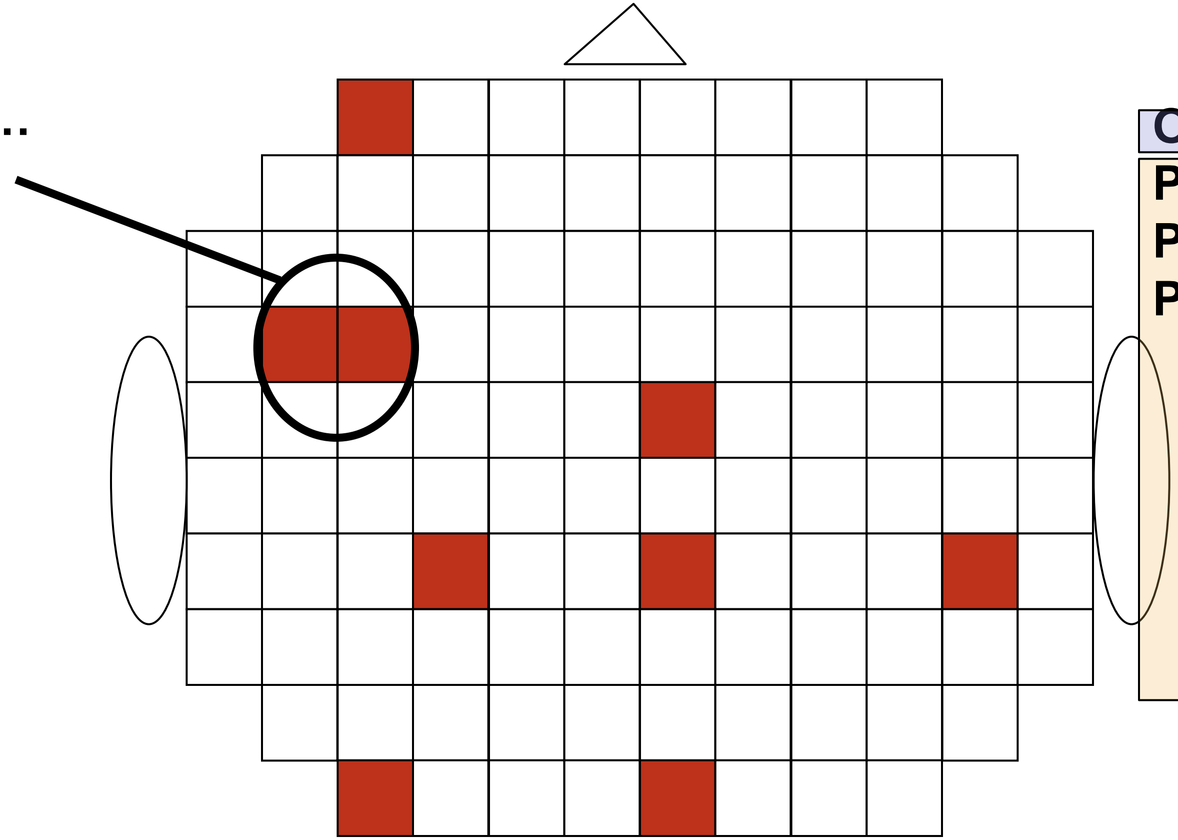
Toy example: 3rd permutation



Observation	**17**
Permutation 1	**13**
Permutation 2	**11**
Permutation 3	**12**

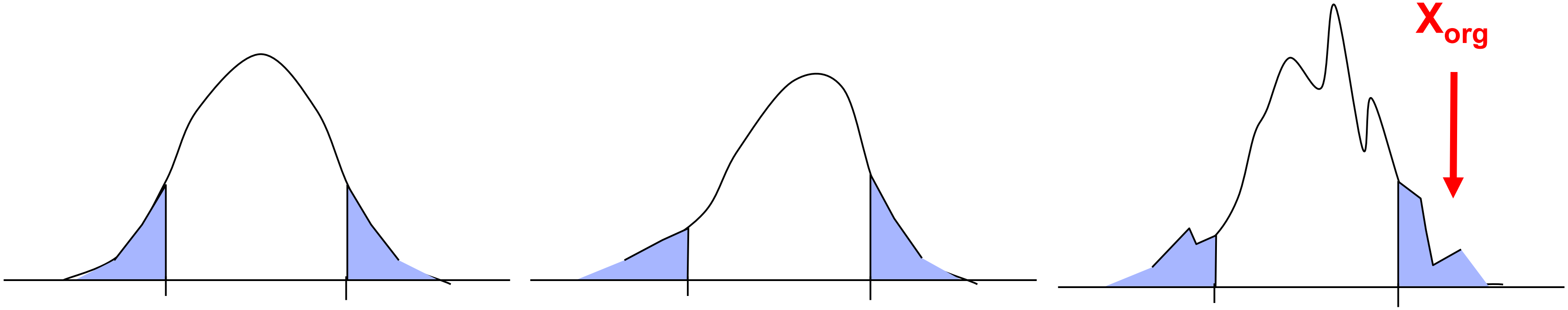
Toy example: N^{th} permutation

statistic = …



Observation	**17**
Permutation 1	**13**
Permutation 2	**11**
Permutation 3	**12**

Assess the likelihood of the *observed max cluster size* given the randomization distribution



Summary statistics

- Parametric statistical test for all channel-time-frequency points
  - probability for H0
  - one H0 for each channel-time-frequency
  - multiple comparison problem
- Non-parametric approach for estimating probability
  - randomization or permutation
  - probability of H0 for arbitrary statistic
  - incorporate prior knowledge in statistic
  - avoid MCP using max statistic

Source-level statistics

Same principles as channel-level statistics

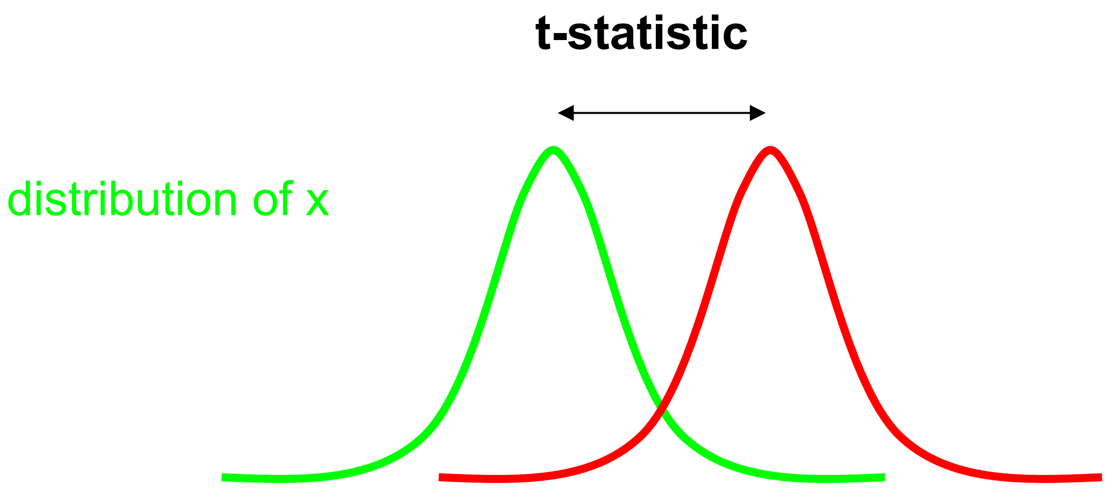
Beamforming

- Swap data between conditions: use common filters

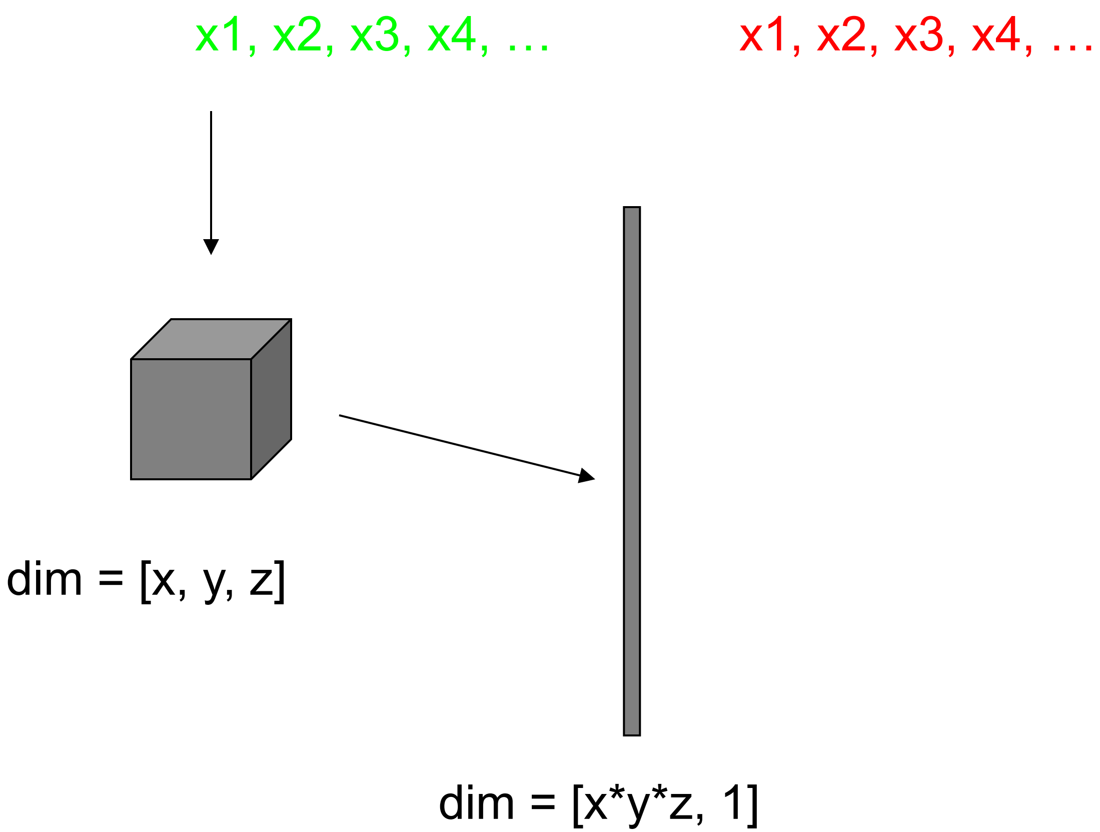
Inferential statistics: parametric

x1, x2, x3, x4, …

x1, x2, x3, x4, …

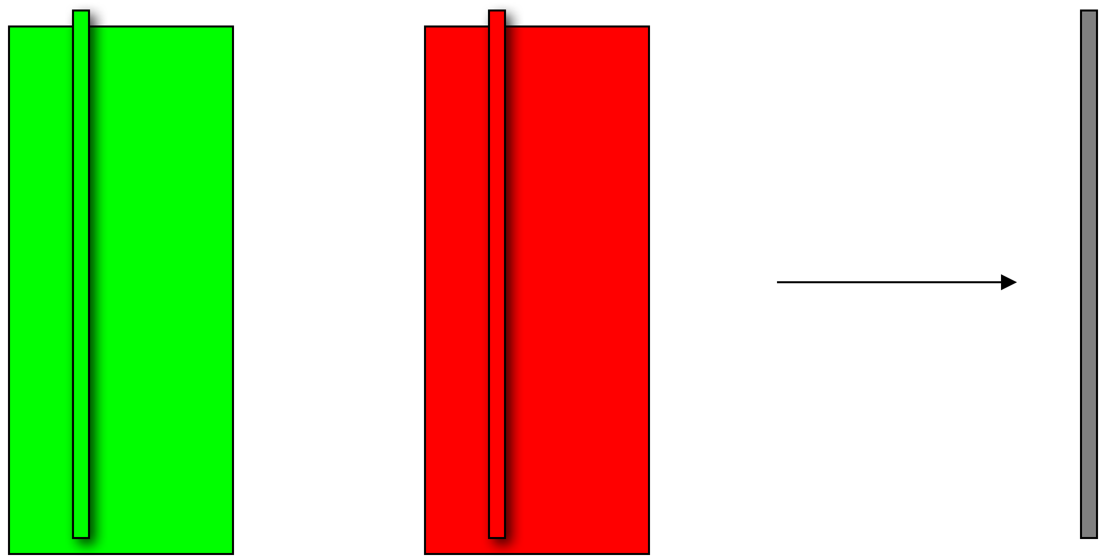


Inferential statistics:
distributed data at source level

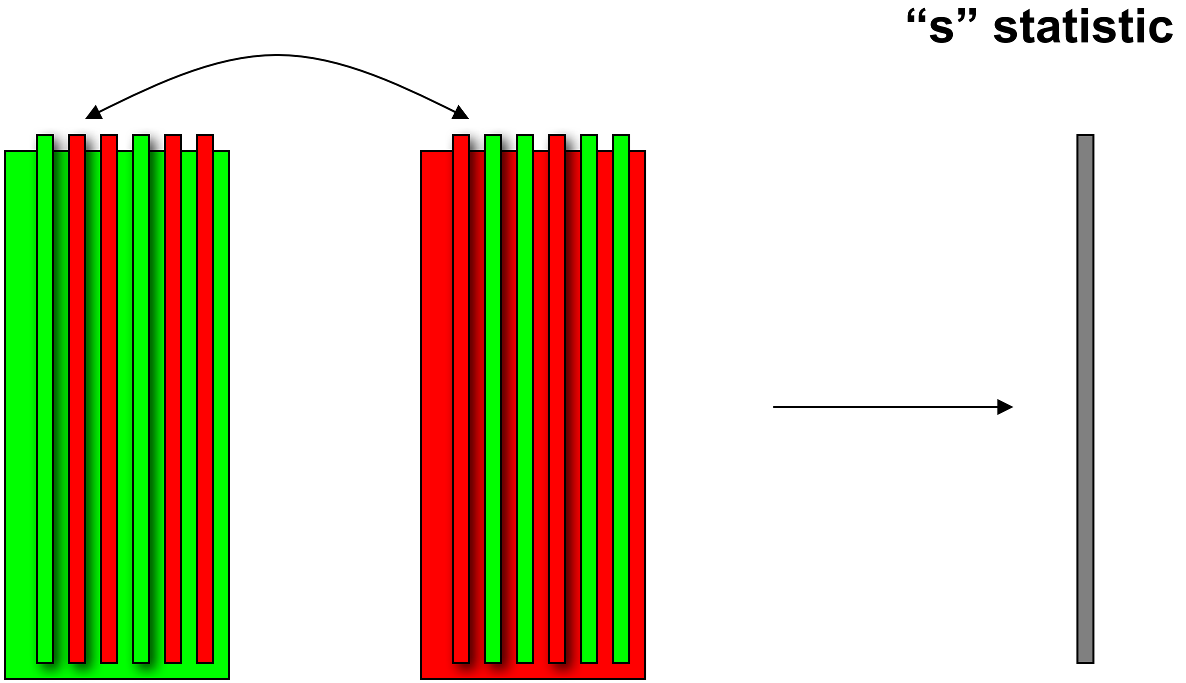


Inferential statistics: parametric

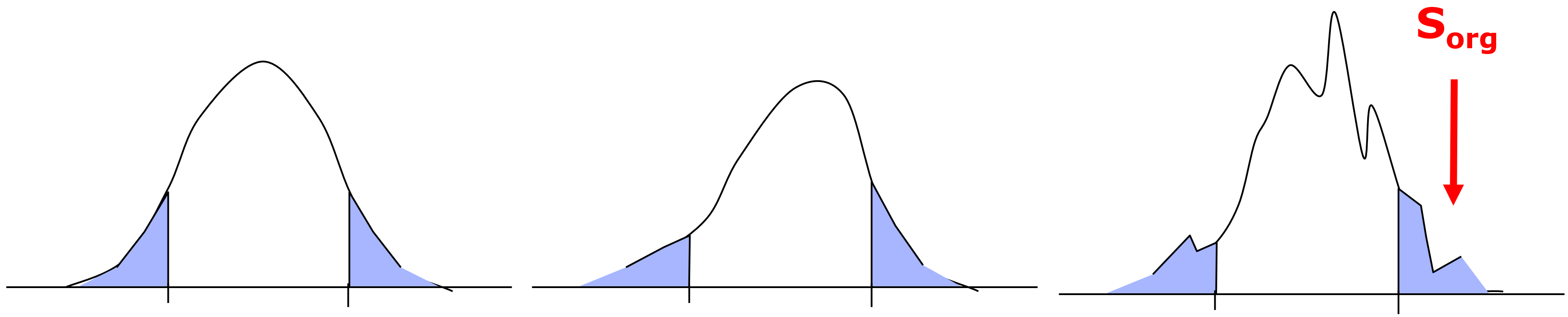
[x1, x2, x3, x4, …   x1, x2, x3, x4, …] → **t-statistic**



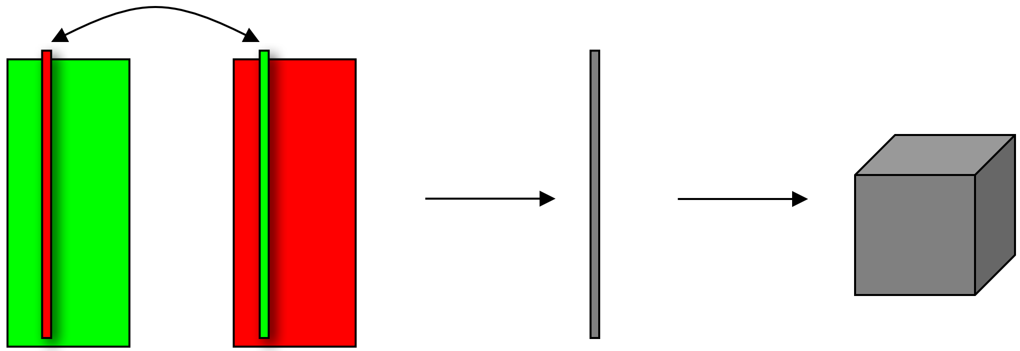
Inferential statistics: permutation approach



Permutation distribution of “s” can take any shape



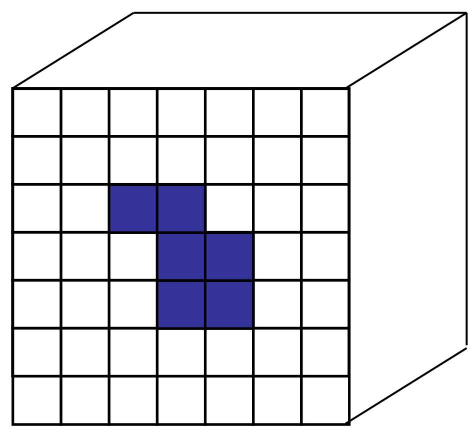
Cluster-based permutation test on source-level

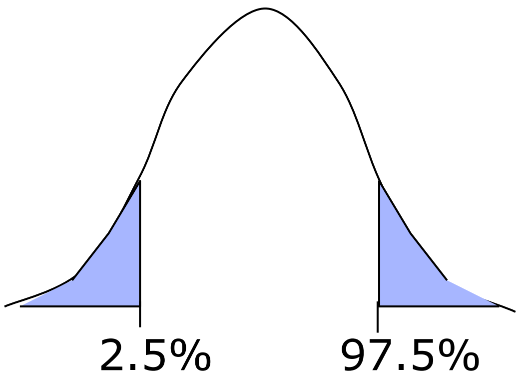


a-priori threshold

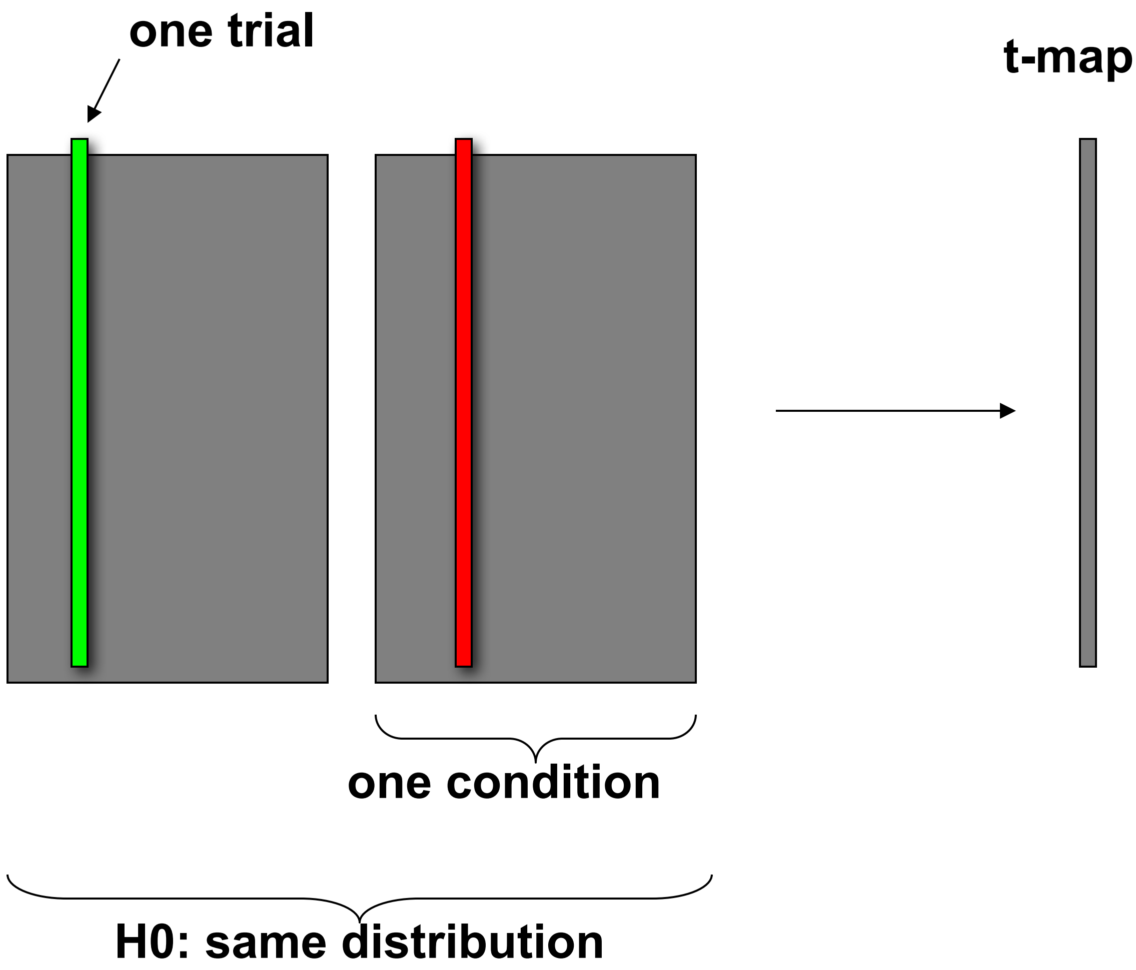
cluster neighbouring voxels

compute sum over cluster





Returning to beamforming



Common filters for beamforming

$$M(t) = G\,X(t) + N$$

$$\hat{X}(t) = W\,M(t) \qquad W = (G^T C^{-1} G)^{-1} G^T C^{-1}$$

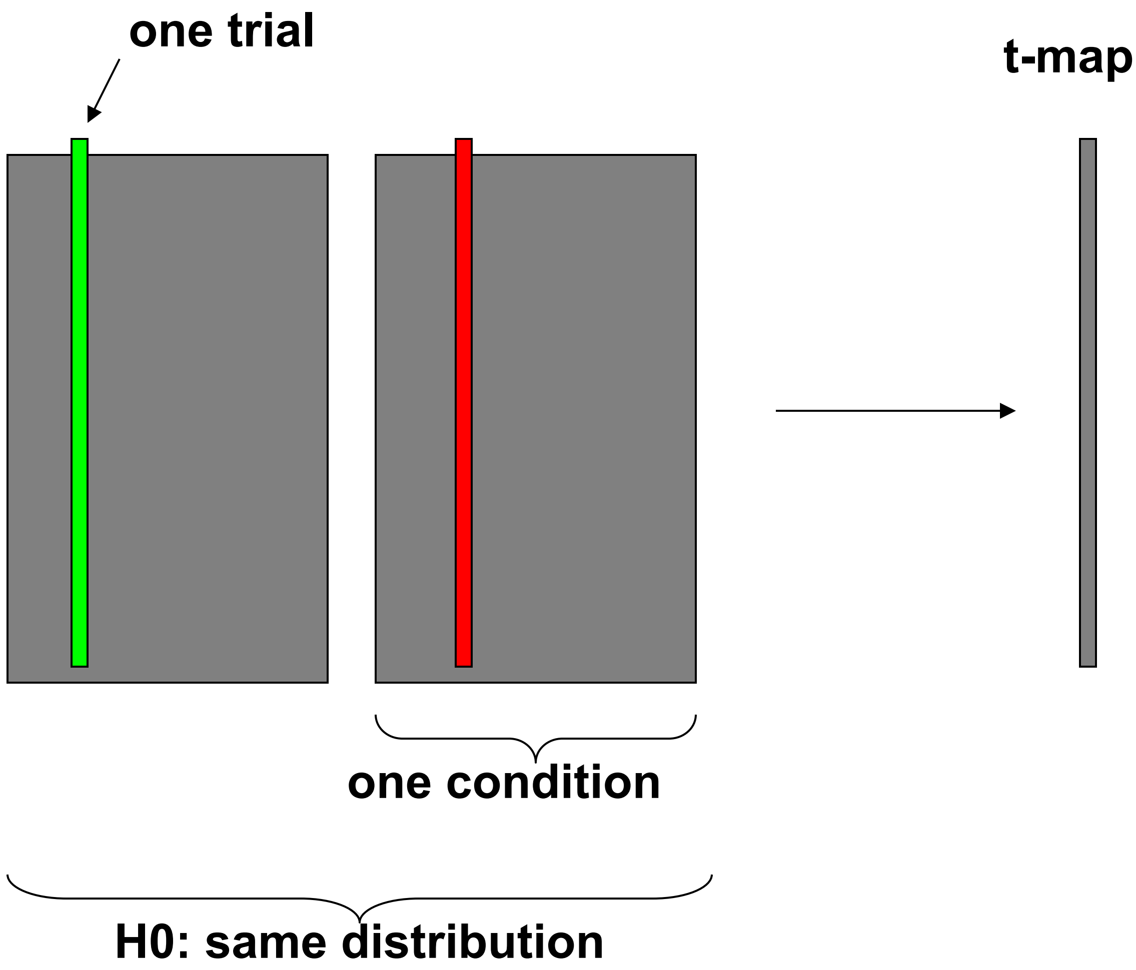
$$P = \hat{X}\,\hat{X}^T = W\,C\,W^T \qquad C = M\,M^T$$

$C_i = M_i\,M_i^T$ trial 1, 2, 3, …

$$C = (C_1 + C_2 + C_3 + \ldots)/n$$

$$P_i = W\,C_i\,W^T$$

Common filters for beamforming



Summary statistics on source-level

- Same principles as for channel-level statistics
- Average covariance over all data for spatial filter estimate
- One spatial filter per voxel
  - common to both conditions
  - single-trial estimates: simple multiplication
- Permutation test not affected
  - exchangeability of data over conditions does not change the optimal filter under H0
  - computationally fast

General summary

A formal hypothesis can be tested with randomization test

- control the chance of false positives
- reduce the false negative rate

Multiple comparison problem

- one hypothesis per channel-time-frequency
- one hypothesis for all data

Increase sensitivity

- using clusters to capture the structure in the data