BIDS stats model

BRAIN IMAGING DATA STRUCTURE

Specify all your analysis in a single file.

Remi Gau
Twitter: @RemiGau

INT
10th November 2022

CPP Lab

Institut de recherche en sciences psychologiques
iPSY
Psychological Sciences Research Institute

UCLouvain

Thanks to Chris Markiewicz and Alejandro de la Vega for letting me reuse some of their slides.

Context

Lack of standardization and poor methods reporting

- hinders reproducibility
- increase inefficiencies

Context

Lack of standardization and poor methods reporting

- hinders reproducibility
- increase inefficiencies

Data and process standardization allow for automation

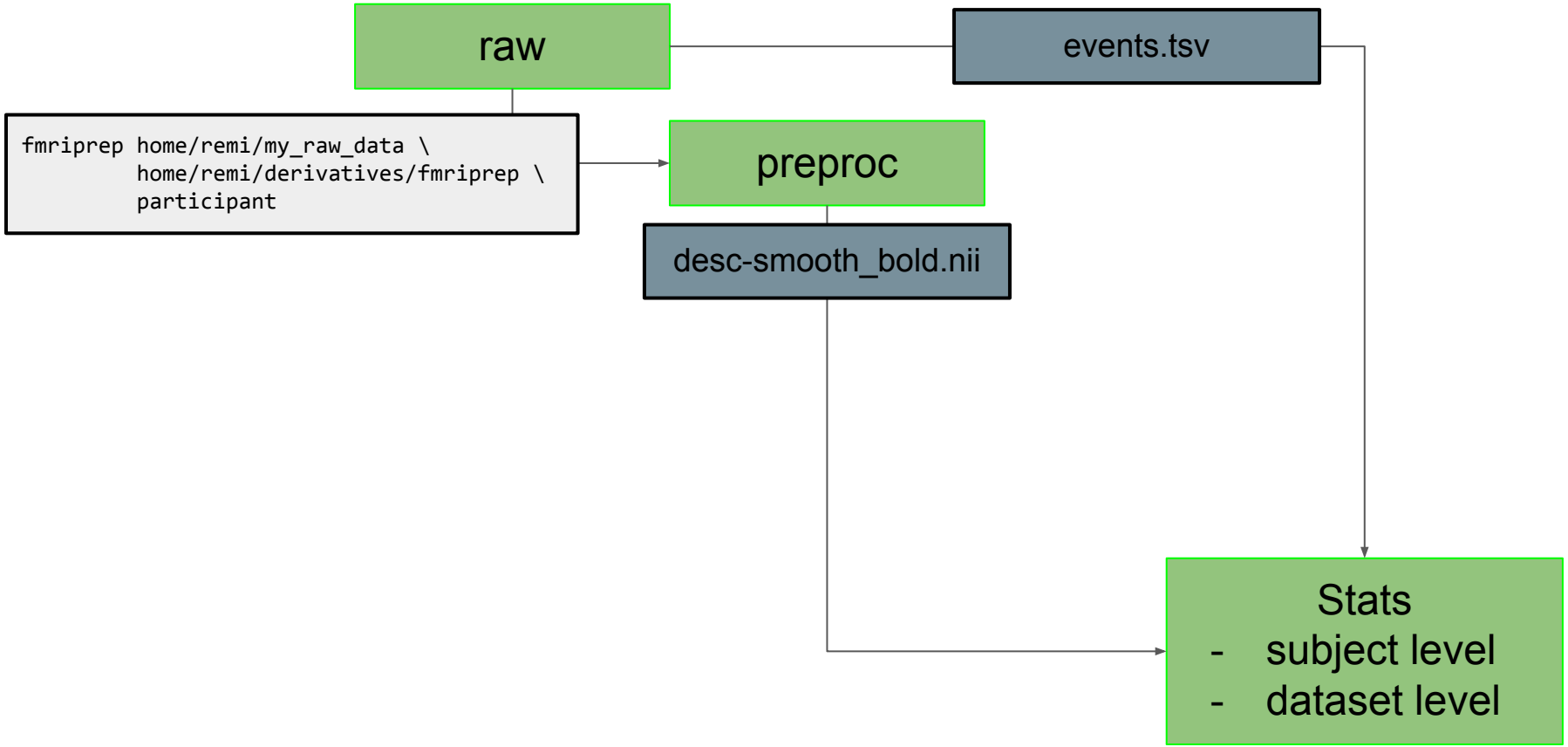
- E.g. BIDS and fmriprep

Problem

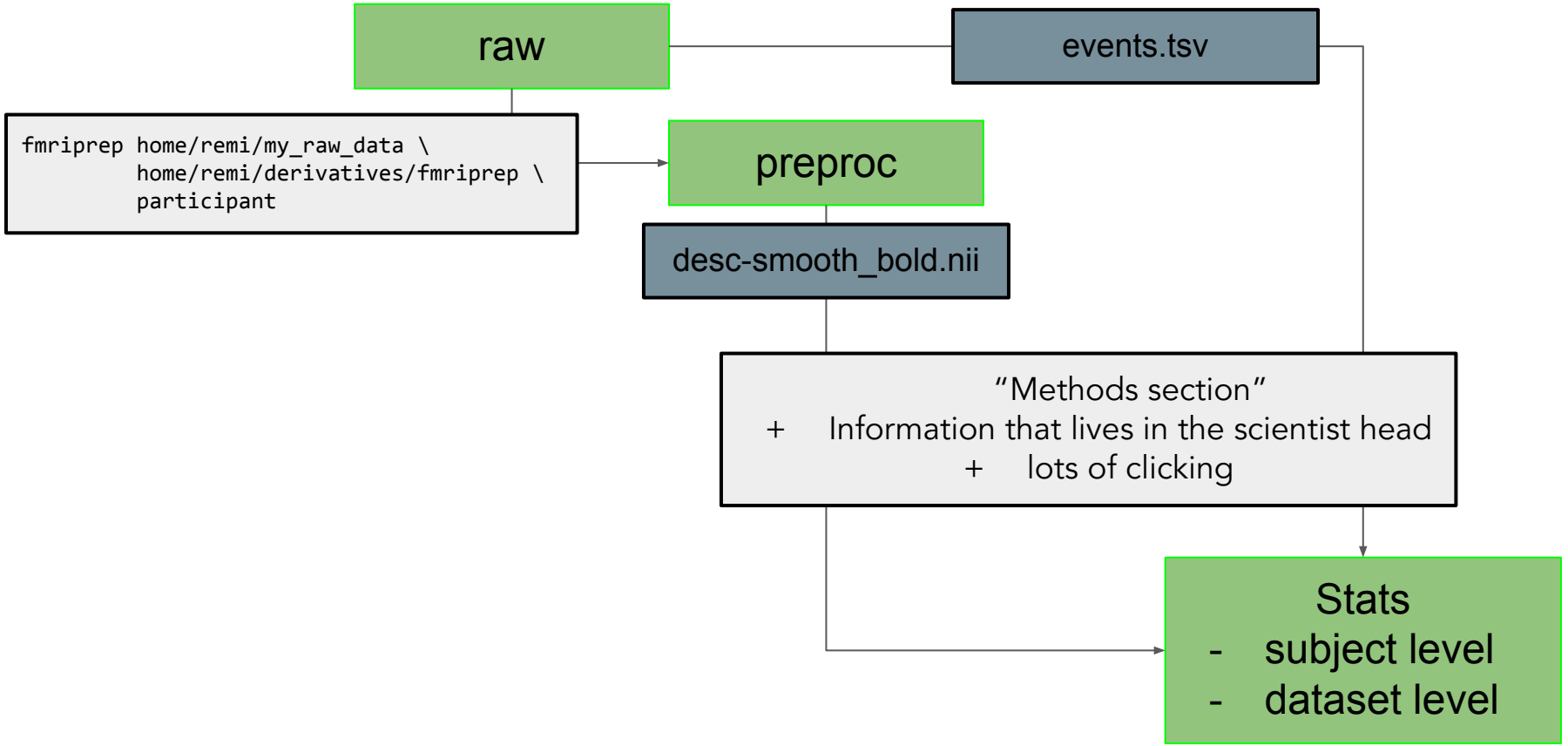
Data and preprocessing standardization are not enough.

- Large-scale efforts fail to reproduce results even with exact data (Botvinik-Nezer et al. 2020)
- fMRI data analysis workflows vary in idiosyncratic ways
- Flexibility in methods: large variety in hypothesis testing outcomes (Carp et al., 2012; Botvinik-Nezer et al., 2020)
- Methods are typically reported in verbal descriptions: difficult to replicate.
- Custom analysis: time-consuming, error-prone, and not accessible

Problem


raw
events.tsv
fmriprep home/remi/my_raw_data \
home/remi/derivatives/fmriprep \
participant
preproc
desc-smooth_bold.nii
Stats
- subject level
- dataset level

Problem


raw
events.tsv
fmriprep home/remi/my_raw_data \
home/remi/derivatives/fmriprep \
participant
preproc
desc-smooth_bold.nii
"Methods section"
+ Information that lives in the scientist head
+ lots of clicking
Stats
- subject level
- dataset level

Problem

Data and preprocessing standardization are not enough.

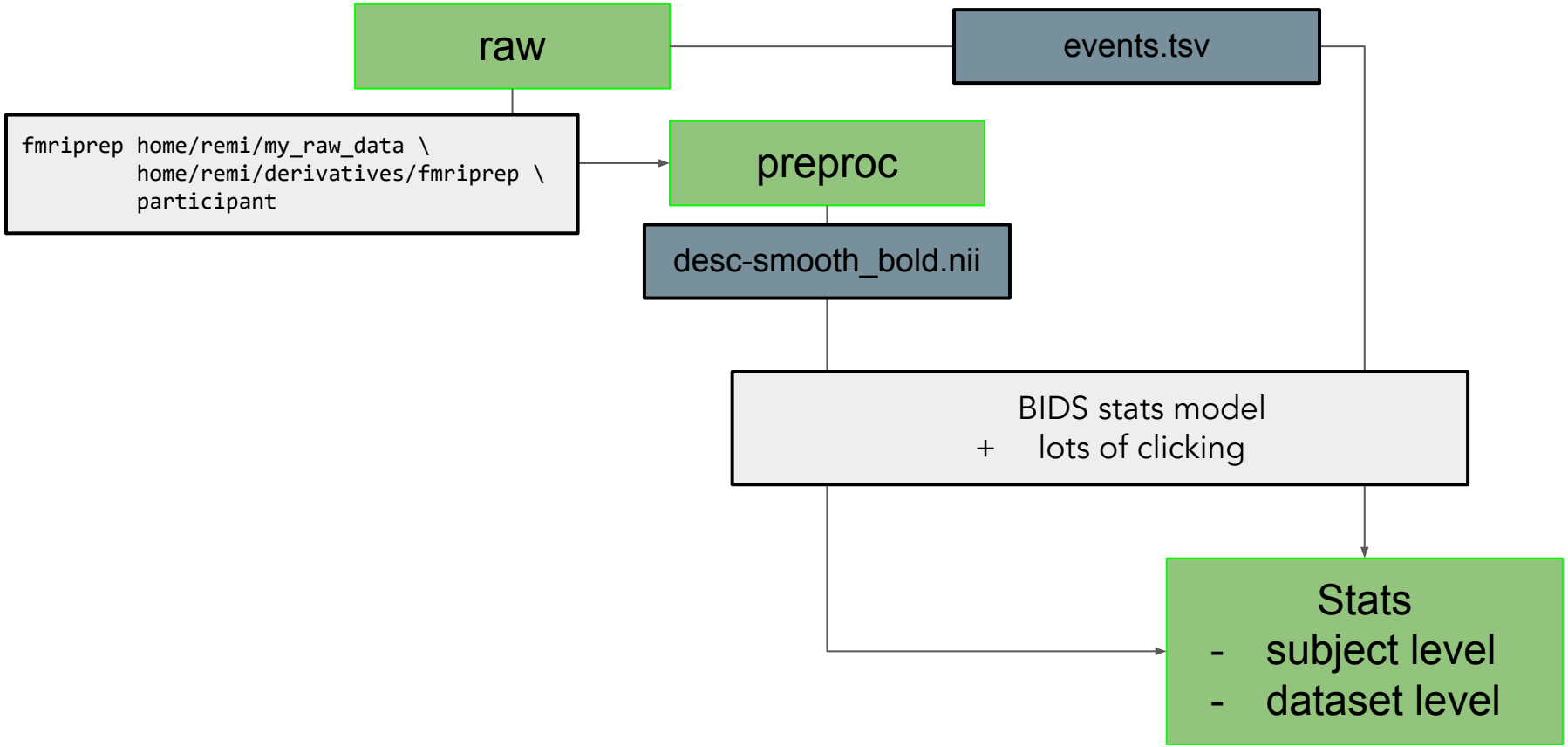
Missing piece: fMRI modeling

Aim

BIDS Stats Models

- describe how to fit statistical models for neuroimaging data
- machine-readable
- prescriptive (as in a recipe)
- sufficient to execute statistical models (given preprocessed BIDS dataset)
- implementation agnostic
- modality agnostic
- minimal configuration and intervention required from the user
- does not require to edit the input datasets

Problem


raw
events.tsv
fmriprep home/remi/my_raw_data \
home/remi/derivatives/fmriprep \
participant
preproc
desc-smooth_bold.nii
BIDS stats model
+ lots of clicking
Stats
- subject level
- dataset level

Aim

Run your entire univariate statistical analysis with:

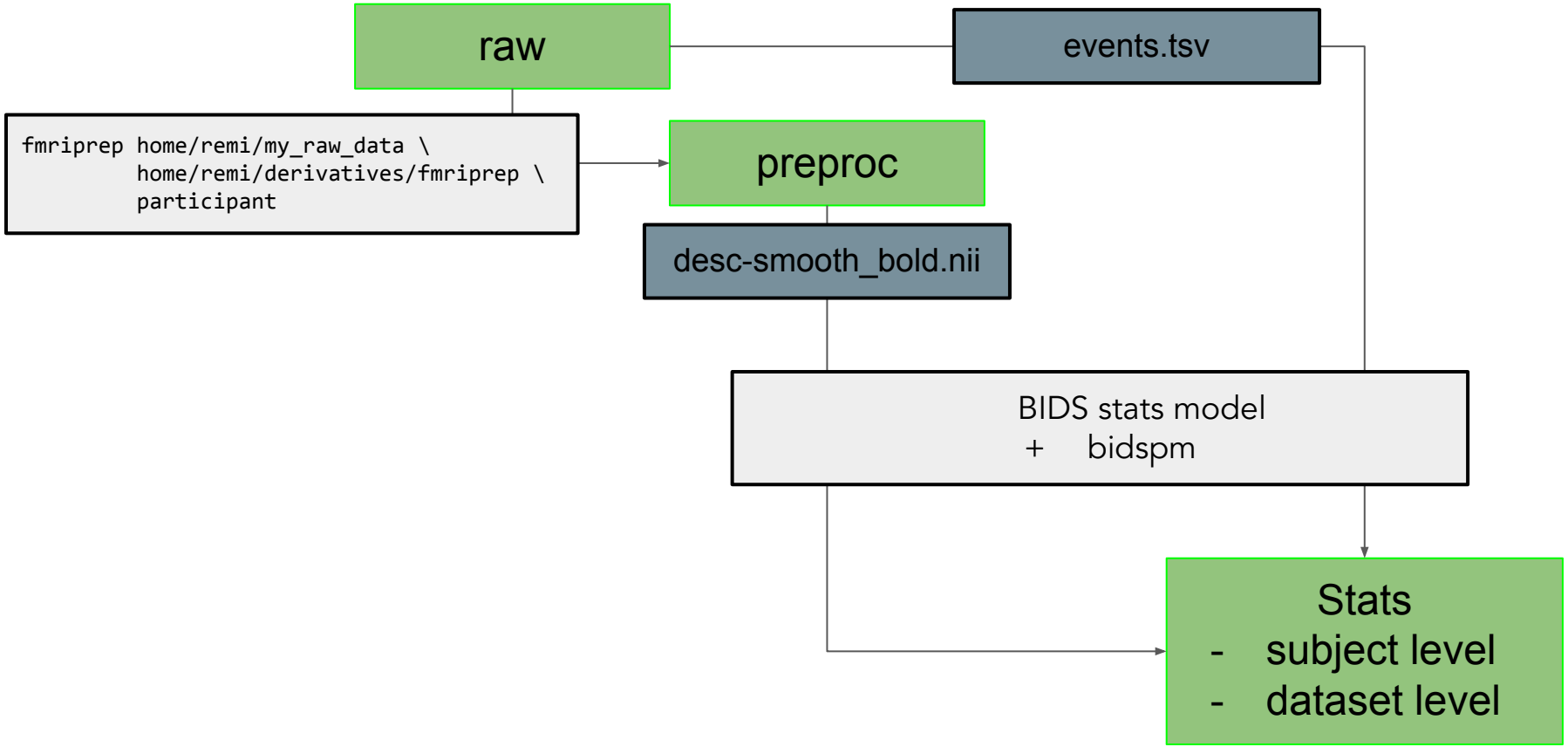
- 50 lines of code
- one JSON file

Implementations

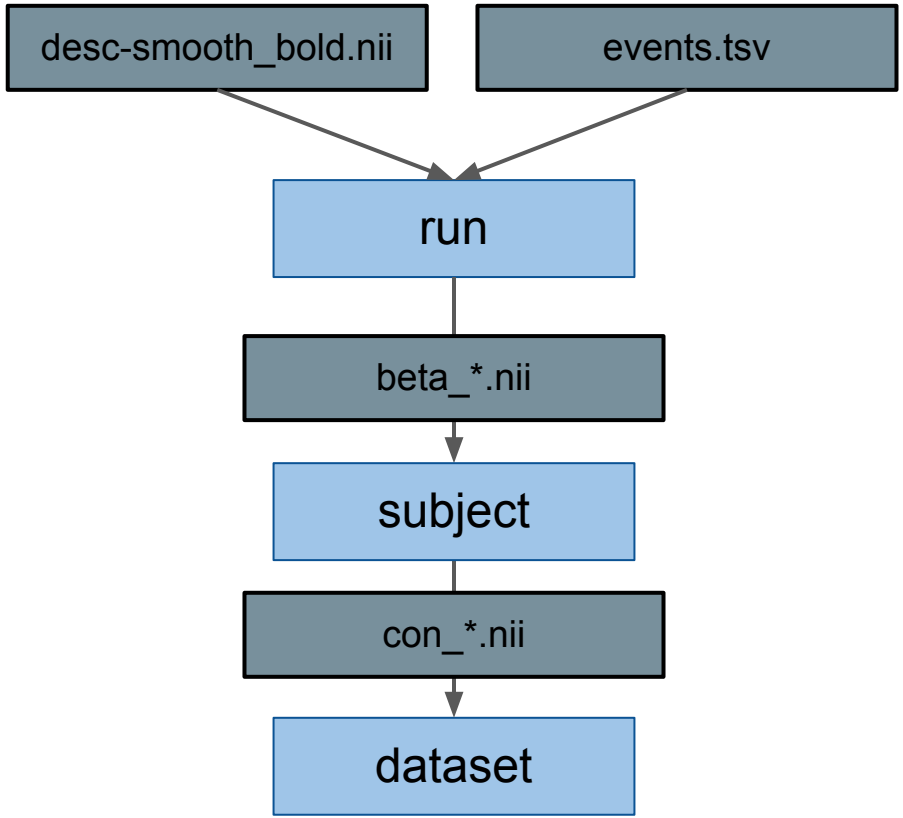
- Modality agnostic in theory
- VERY fmri centric in practice (for now)

- Actual implementations
  - Neuroscout (web interface, naturalistic stimuli datasets)
  - Fitlins (python)
  - Bidspm (matlab / octave, SPM)

Implementations

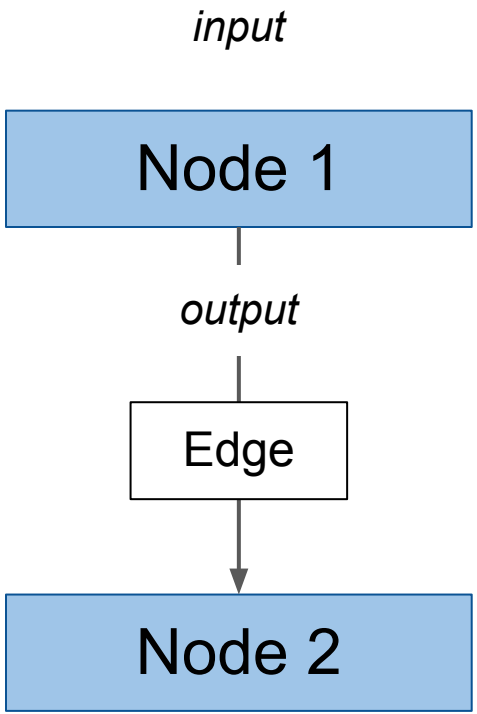


Mass univariate analysis (SPM)

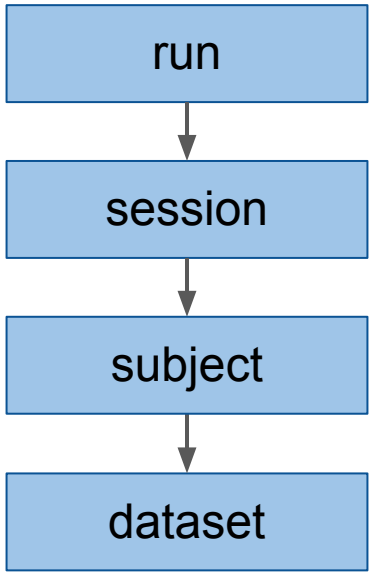


BIDS stats model - DAG

- Directed acyclic graph

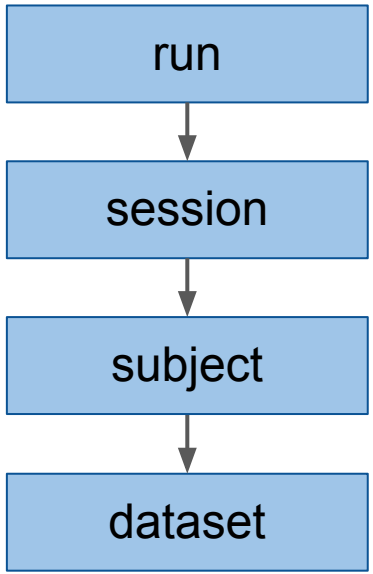


BIDS stats model - DAG



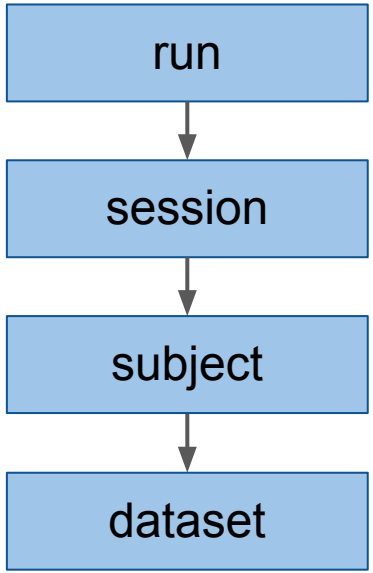
BIDS stats model - DAG

Basic summary statistics with four nodes

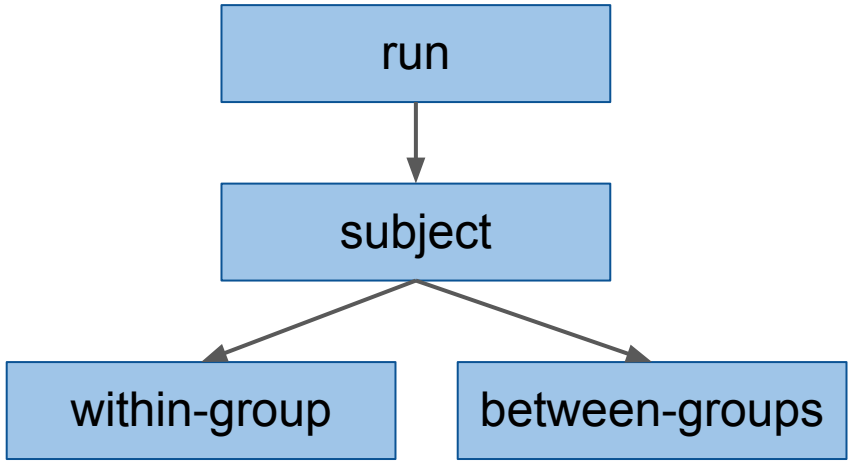


BIDS stats model - DAG

Basic summary statistics with four nodes



Multiple dataset-level nodes from the same run and subject level estimates.



BIDS stats model - overview

BIDS stats models are a JSON file with 3 components:

1. Basic metadata

2. Input selectors
   a. BIDS entities and metadata

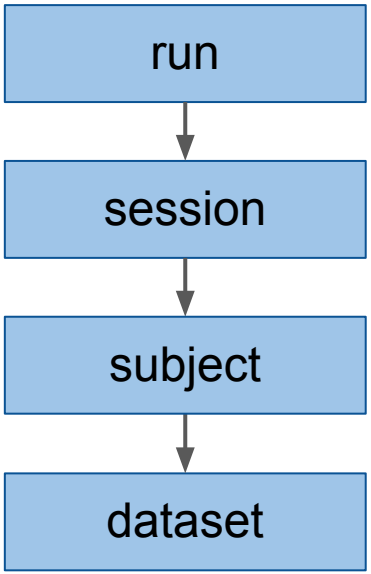
3. Computational graph
   a. Nodes correspond to estimators
   b. Edges connect nodes

```
{
  "Name": "MyModel",
  "BIDSModelVersion": "1.0.0",
  "Description": "Simple motor model",

  "Input": {"task": "motor"},

  "Nodes": [...],
  "Edges": [...]
}
```

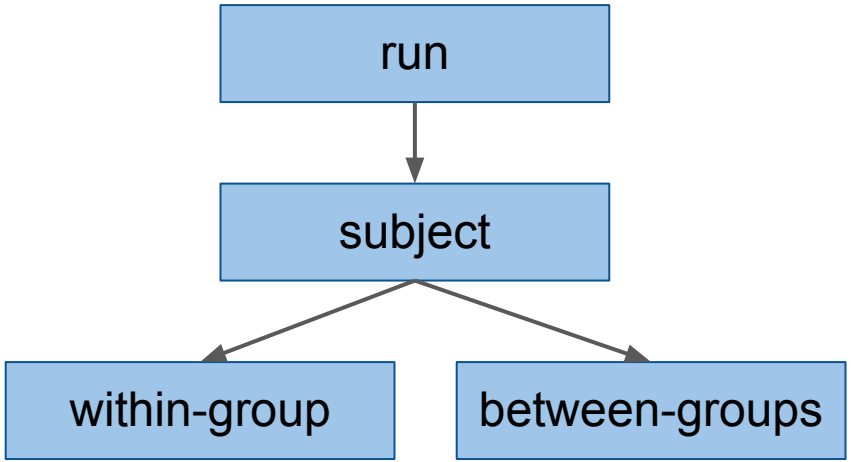
BIDS stats model - edges




```
{
  "Nodes": [
    {"Name": "run", …},
    {"Name": "session", …},
    {"Name": "subject", …},
    {"Name": "dataset", …}
  ],
  "Edges": [
    {"Source": "run", "Destination": "session"},
    {"Source": "session", "Destination": "subject"},
    {"Source": "subject", "Destination": "dataset"}
  ]
}
```

BIDS stats model - edges

```
{
  "Nodes": [
    {"Name": "run", ...},
    {"Name": "subject", ...},
    {"Name": "within-group", ...},
    {"Name": "between-groups", ...}
  ],
  "Edges": [
    {"Source": "run", "Destination": "subject"},
    {"Source": "subject",
     "Destination": "within-group"},
    {"Source": "subject",
     "Destination": "between-groups"}
  ]
}
```



BIDS stats model - Nodes

Inputs

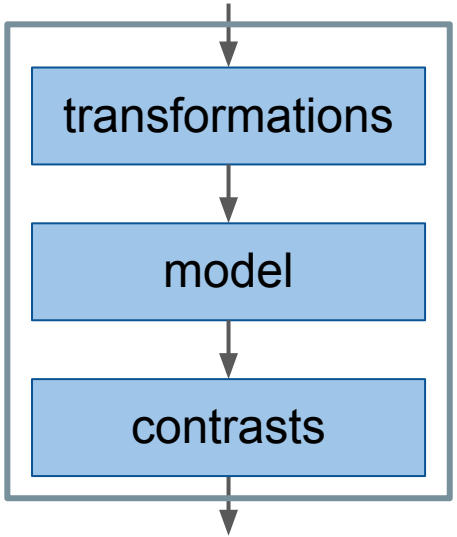
- Variables from input dataset(s)
- Images
  - Run-level: BOLD series
  - Higher-level: Contrast maps

Transformations

- Manipulate input variables:
  - Example: Filter certain rows of events.tsv

Outputs

- Images: Contrast maps



BIDS stats model - Nodes

Nodes properties:

- Level:
  - "Run", "Session", "Subject" or "Dataset"
- Name: Unique name for edges
- GroupBy:
  - Variables that identify subsets of inputs
  - Easier (for me) to think of as splitting
  - Similar to the GroupBy operation in R or pandas dataframes.

```
{
  "Level": "Run",
  "Name": "Run",
  "GroupBy": ["run", "subject"],
  "Model": {...},
  "Contrasts": [...],
}
```

BIDS stats model - design matrix

Given the variables available in:

- events.tsv (raw)
- timeseries.tsv (derivative)

We define a design matrix in `Model.X`.

HRF parameters:

- What to convolve
- With what HRF

Options:

- high pass filter,
- inclusive mask

```
{
  "Name": "Run",

  "Model": {

    "X": ["ev1", "ev2", "confound1", "confound2", 1]

    "HRF": {
         "Variables": ["ev1", "ev1"],
         "Model": "spm"
    },

    "Options": {
            "HighPassFilterCutoffHz": 0.0078,
            "Mask": {"suffix": ["mask"], "desc": ["brain"]}
    },

    "Software": {
            "SPM": {
             "SerialCorrelation": "AR(1)"
            }
    }

  }
}
```

BIDS stats model - contrasts

Contrasts are the output of a node.

The `Contrasts` list is a weighted sum of betas for t contrasts (F contrast supported too).

To output individual betas, the `DummyContrasts` object constructs a contrast with one condition and weight `[1]` for each beta.

```
{
  "Name": "Run",
  ...
  "Contrasts": [
    {
      "Name": "ev1_vs_ev2",
      "ConditionList": ["ev1", "ev2"],
      "Weights": [1, -1],
      "Test": "t"
    }
  ],
  "DummyContrasts": {
    "Conditions": ["ev1", "ev2"],
    "Test": "t"
  }
  ...
}
```

BIDS stats model - subject level

Average beta images from the run level.

```
{
  "Level": "Subject",
  "Name": "subject_level",
  "GroupBy": [
    "contrast",
    "subject"
  ],
  "Model": {
    "X": [
      1
    ],
    "Type": "glm"
  },
  "DummyContrasts": {
    "Test": "t"
  }
}
```

BIDS stats model - dataset level

Average across all subjects

```
{
  "Level": "Dataset",
  "Name": "dataset_level",
  "GroupBy": [
    "contrast"
  ],
  "Model": {
    "X": [
      1
    ],
  },
  "DummyContrasts": {
    "Test": "t"
  }
}
```

Average by Group

```
{
  "Level": "Dataset",
  "Name": "within_group",
  "GroupBy": [
    "group"
    "contrast"
  ],
  "Model": {
    "X": [
      1
    ],
  },
  "DummyContrasts": {
    "Test": "t"
  }
}
```

2 samples T-test

```
{
  "Level": "Dataset",
  "Name": "between_groups",
  "GroupBy": [
       "Contrast"
       ],
  "Model": {
       "X": [
      1,
      "group"
      ]
  },
  "Contrasts": [
    {
     "Name": "blind_gt_control",
     "ConditionList": [
        "Group.blind",
        "Group.control"
     ],
     "Weights": [
        1,
        -1
     ],
     "Test": "t"
    }
  ]
}
```

BIDS apps CLI

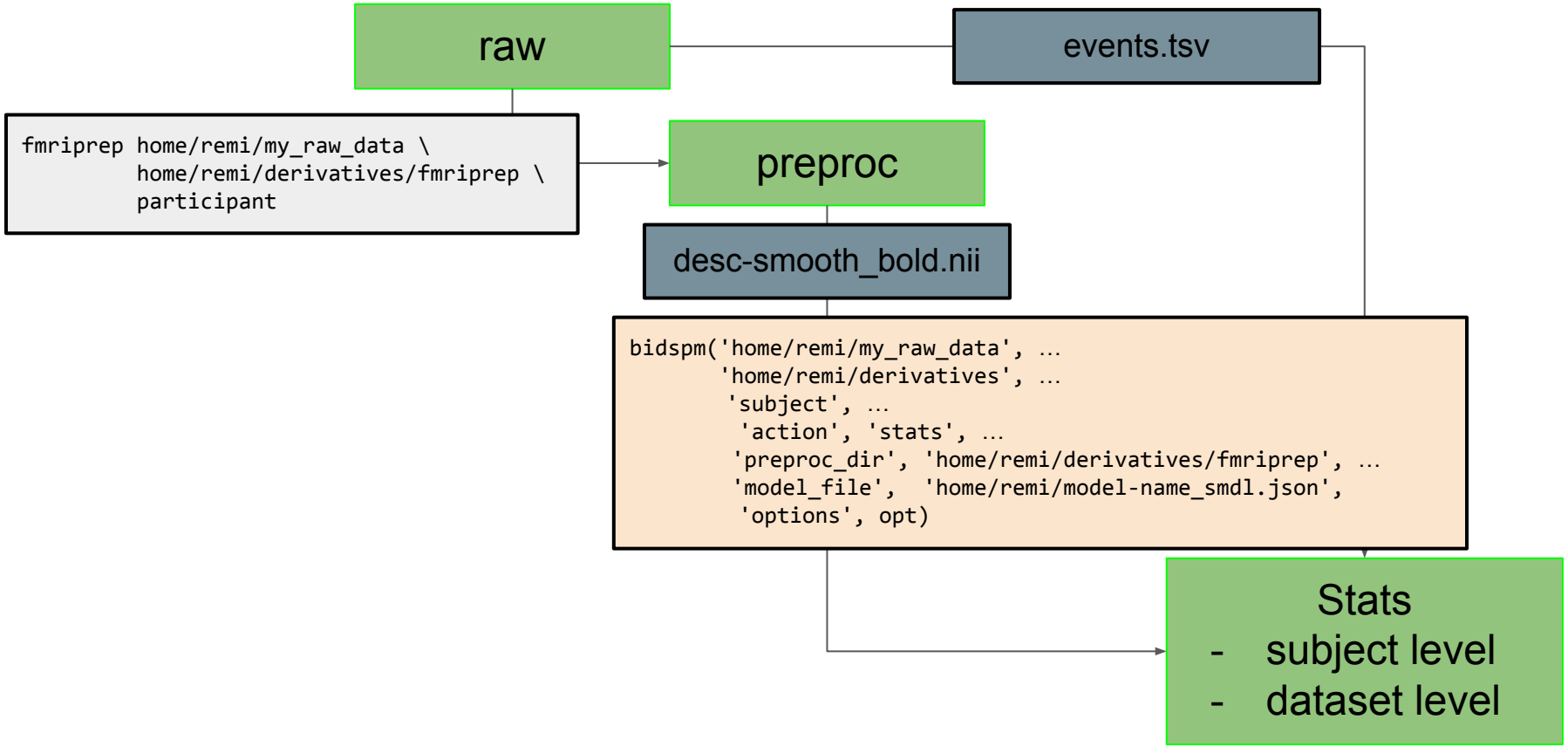
```
bids-app /bids-directory /output-directory participant [OPTIONS]
```

```
fmriprep /data/raw /data/processed/ participant --participant-label pixar001 [OPTIONS]
```

```
fitlins /data/raw /data/stats dataset --derivatives /data/processed/fmriprep --model model-name_smdl.json [OPTIONS]
```

```
bidspm(raw_data, output, 'subject', …
        'pariticipant_label', {'01', '02'}
        'action',      'stats', ...
        'preproc_dir', bidspm_preproc, ...
        'model_file',  'home/remi/model-name_smdl.json',
        'options', opt)
```

Implementations



BIDS stats model - advanced

- What HRF to use?
- What type of confounds to include?
- Should I scrub my data?

Choose a model in principled manner without data peeking / double-dipping / p-hacking.

Bayesian model comparison (MACS toolbox)

BIDS stats model - advanced

Bayesian model comparison (MACS toolbox)

12 different BIDS stats models

HRF

- HRF
- HRF + temporal
- HRF + temporal + dispersion

Confounds

- None
- CSF + WM

Scrubbing

- no outlier removal
- with outlier removal

```
opt = opt_stats_subject_level();

models = opt.toolbox.MACS.model.files

for i = 1:numel(models)
  opt.model.file = models{i};
  bidsFFX('specify', opt);
end

bidsModelSelection(opt, 'action', 'all');
```

BIDS stats model - advanced

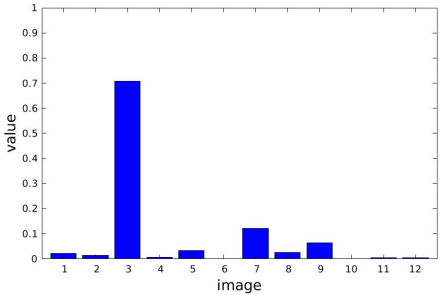
Bayesian model comparison (MACS toolbox)

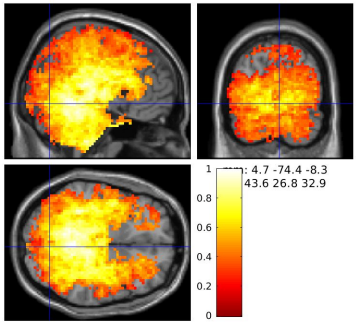
moddel 03:

No Derivative

With Tissue Confounds

No Scrubbing



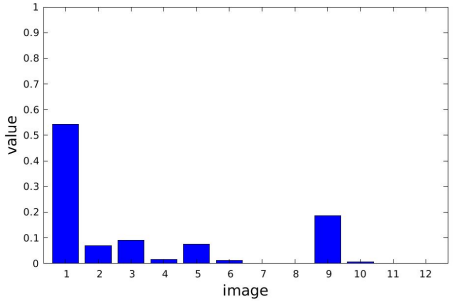


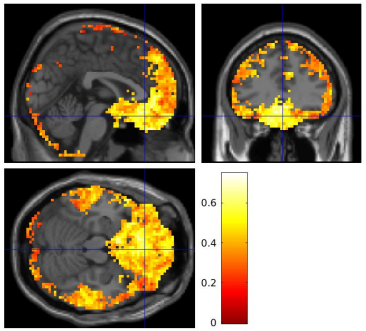
model 01:

No Derivative

No Tissue Confounds

No Scrubbing





BIDS stats model - demo

- Set up
  - Grab dataset from openneuro
  - Copy preprocessed BOLD files and smooth them

- Create default model
  - Validate it: Validator

- Run BIDS app command

BIDS stats model - resources

BIDS stats models and where to find them

- Zoo
- demos
- tests

Specification:

- Walkthrough
- JSON 101

Variable transformations specification:

- bids-matlab
- Pybids-transforms-v1