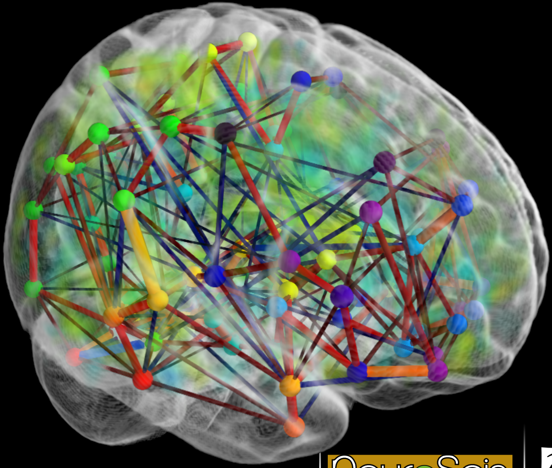
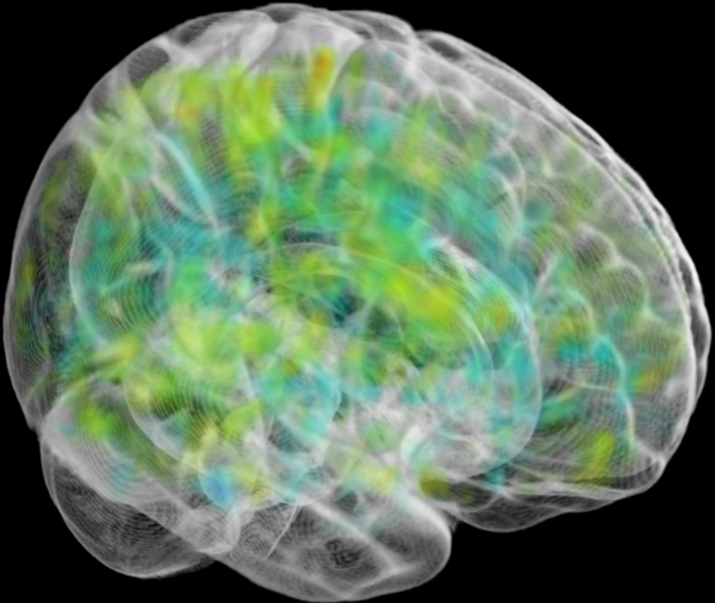
Modeling spontaneous brain activity in Python
Scientific progress and software challenges

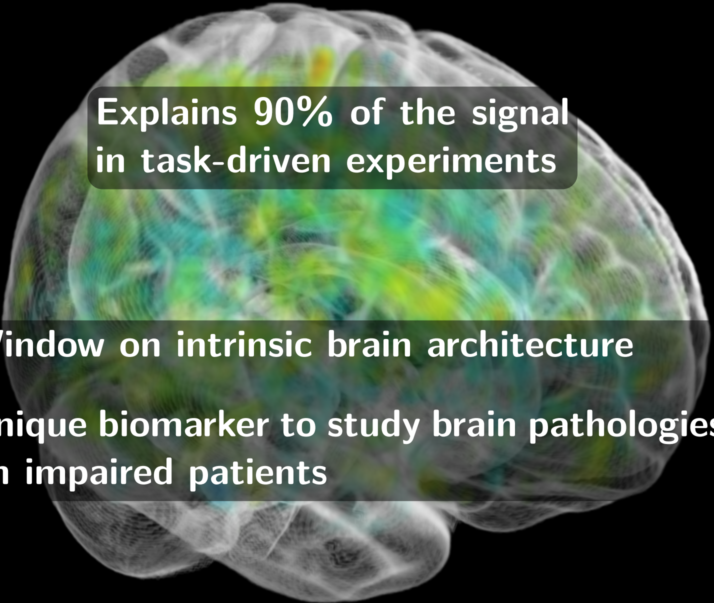
Gaël Varoquaux, INRIA and Neurospin



Spontaneous brain activity

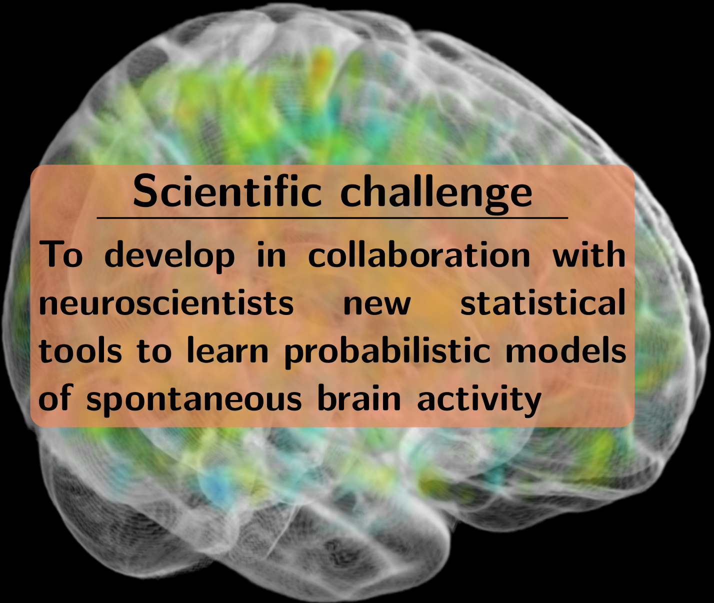


Why study spontaneous brain activity?



Explains 90% of the signal
in task-driven experiments

- Window on intrinsic brain architecture
- Unique biomarker to study brain pathologies on impaired patients



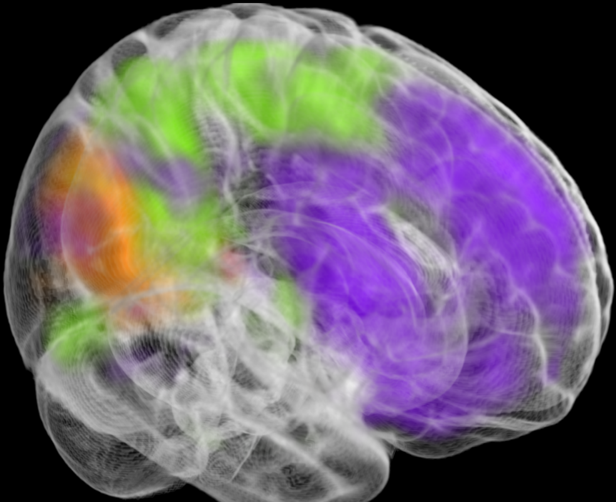
Scientific challenge

To develop in collaboration with neuroscientists new statistical tools to learn probabilistic models of spontaneous brain activity

Outline

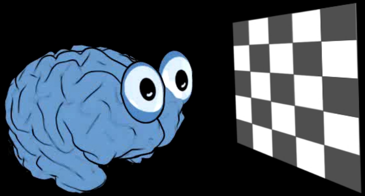
1. **Spatial patterns of brain activity**
2. **Beyond activation maps**
3. **Inter-subject comparisons**
4. **From models to software tools?**

1 Spatial patterns of brain activity



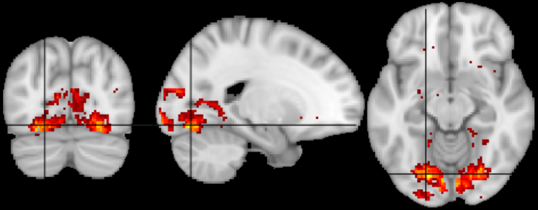
1 Conventional brain mapping

- Study of stimuli response



- Mass-univariate statistics:
  for each voxel $\mathbf{X} = \beta \mathbf{Y} + \mathbf{E}$

- Group inference: subject-variability model on β



1 Conventional brain mapping – software

Nipy: NeuroImaging in Python
Berkeley, Stanford, Neurospin . . .

Vision: Open code shared between labs

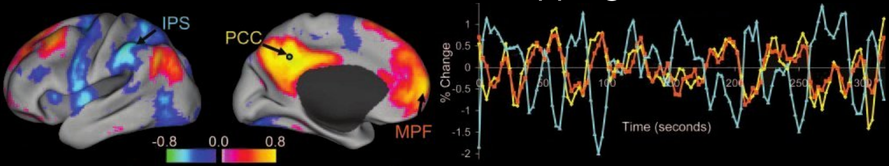
Progress:
- Statistical models implemented
- API difficult to use
- Good Input/Output code
- Preprocessing not implemented

Roadblocks:
- Different teams $\Rightarrow$ different visions
- Scientists can't justify time on "solved problems"

1 Spatial correlation maps of spontaneous activity

- Biswal 1995: strong correlation between activity in left and right motor cortex at rest

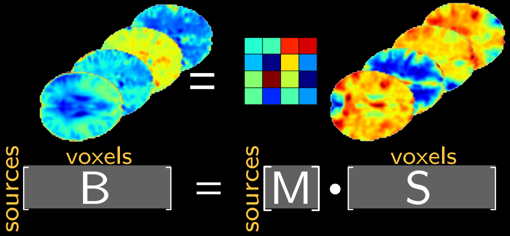
- Later: seed-based correlation mapping



The human brain is intrinsically organized into dynamic, anticorrelated functional networks (Fox 2005)

How many? How to choose seeds?

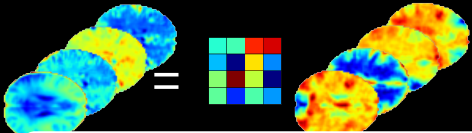
1 Independent component analysis



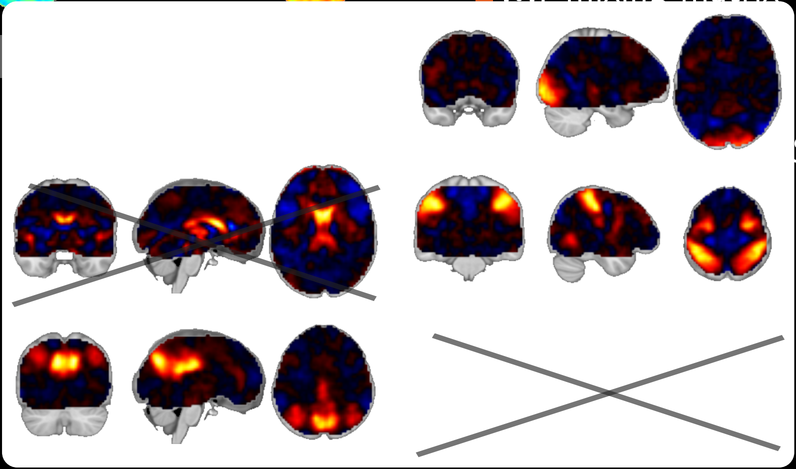
- **B**: observed images
- **M**: mixing matrix
- **S**: sources

- Minimize mutual information between patterns S.

1 Independent component analysis

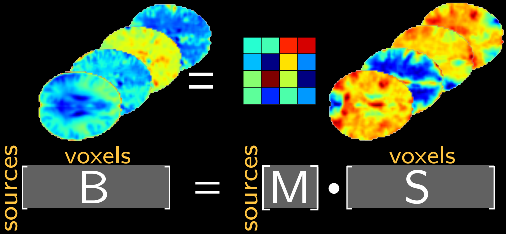


- **B**: observed images
- **M**: mixing matrix



S.

Independent component analysis



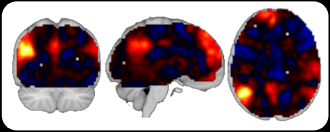
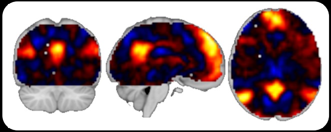
- **B**: observed images
- **M**: mixing matrix
- **S**: sources

- Minimize mutual information between patterns S.

No noise model

$\Rightarrow$ **Lack of reproducibility** $+$ **Fits noise**

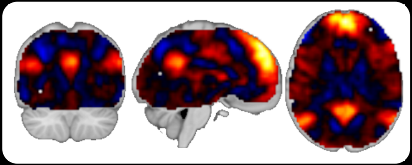
1 Model subject-to-subject variability

$\mathbf{A}_1$  $\neq$  $\mathbf{A}_2$

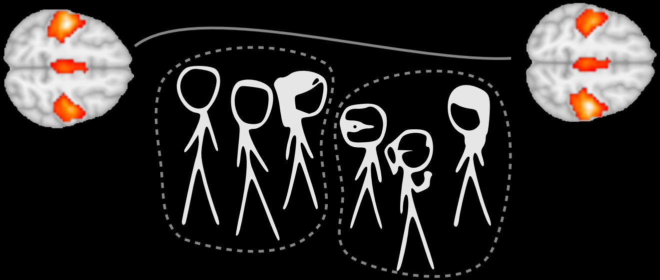
Multivariate random effects model:

$$\begin{aligned} \mathbf{Y}_s &= \text{loadings} \times \mathbf{P}_s + \text{intra-subject noise} && \text{PCA} \\ \{\mathbf{P}_s\} &= \text{loadings} \times \mathbf{B} + \text{inter-subject variability} && \text{CCA} \\ \mathbf{B} &= \mathbf{M} \times \mathbf{A} && \text{ICA} \end{aligned}$$

$\Rightarrow$ **Group-level networks**



1 Model subject-to-subject variability



Reproducibility across random groups

	no CCA	CCA + ICA
Subspace	.36 (.02)	.71 (.01)
One-to-one	.36 (.02)	.72 (.05)

Varoquaux, NeuroImage 2010

1 Efficient Python implementation (CanICA)

Problem to solve:

(1) $\mathbf{Y_s} = \text{loadings} \times \mathbf{P_s} + \ldots$ PCA: SVD

(2) $\{\mathbf{P_s}\} = \text{loadings} \times \mathbf{B} + \ldots$ CCA: SVD

(3) $\mathbf{B} = \mathbf{M} \times \mathbf{A}$ ICA: iterations

$+$ Recomputed many times across random groups

Step 2 and 3: Small data size $\Rightarrow$ not bottleneck

Step 1: ■ Independent problems per subject

$\Rightarrow$ Parallel runs and caching of the results

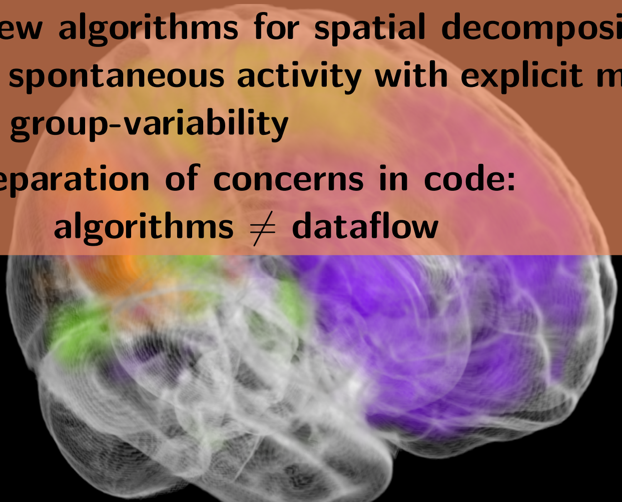
Joblib: Python functions as pipeline jobs

Goals: remove dataflow and persistence problems from algorithmic code

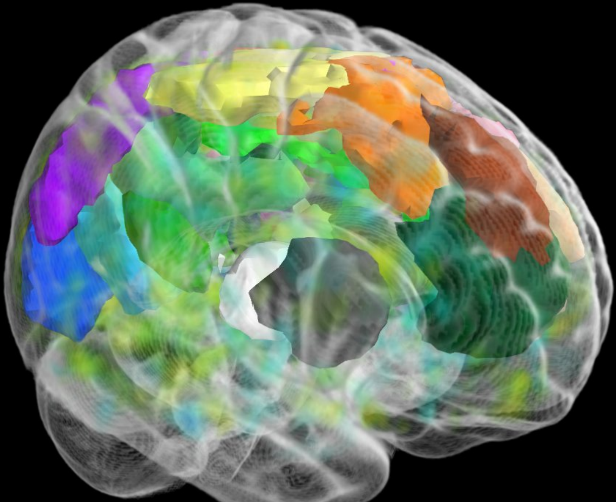
Spatial patterns of brain activity

- New algorithms for spatial decomposition of spontaneous activity with explicit model of group-variability
- Separation of concerns in code:

  algorithms $\neq$ dataflow



2 Beyond activation maps



2 Segmenting sparse regions

$$\begin{array}{cc} & \text{voxels} \\ \text{sources} & \begin{bmatrix} \quad B \quad \end{bmatrix} \end{array} = \begin{array}{cc} & \\ \text{sources} & \begin{bmatrix} M \end{bmatrix} \end{array} \bullet \begin{array}{c} \text{voxels} \\ \begin{bmatrix} \quad S \quad \end{bmatrix} \end{array}$$

2 Segmenting sparse regions

$$\cancel{\text{sources}\overset{\text{voxels}}{\left[\quad B \quad\right]} = \text{sources}\left[M\right]\cdot\overset{\text{voxels}}{\left[\quad S \quad\right]}}$$

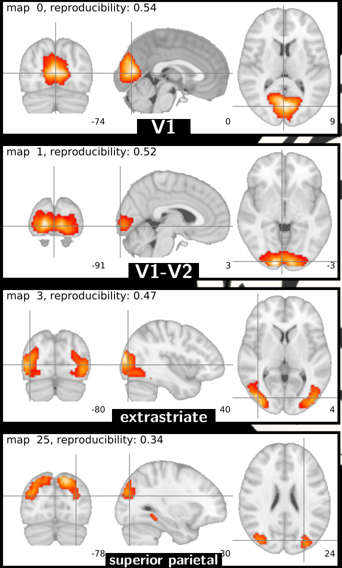
$$\text{sources}\overset{\text{voxels}}{\left[\quad B \quad\right]} = \text{sources}\left[M\right]\cdot\left(\overset{\text{voxels}}{\left[\quad S \quad\right]} + \overset{\text{voxels}}{\left[\quad Q \quad\right]}\right)$$

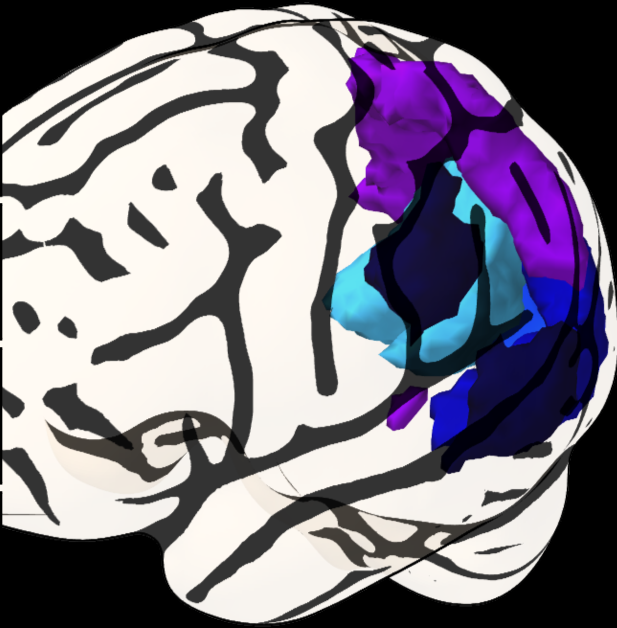
- Interesting sources S **sparse**
- Q: Gaussian noise

$\Rightarrow$ Null hypothesis: centered normal distribution.

2 A full-brain parcellation

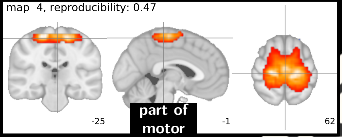
Visual system

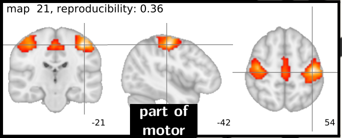


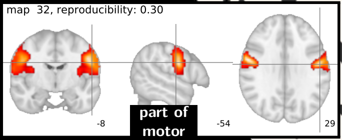


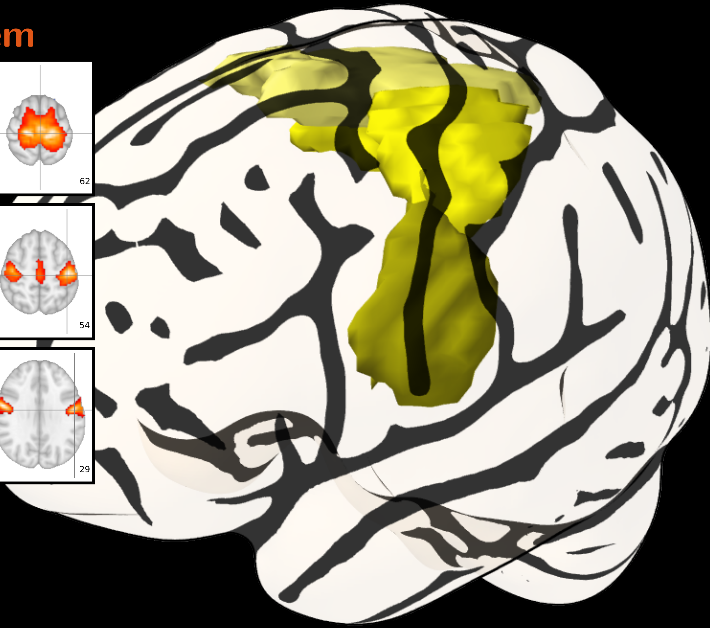
A full-brain parcellation

Motor system

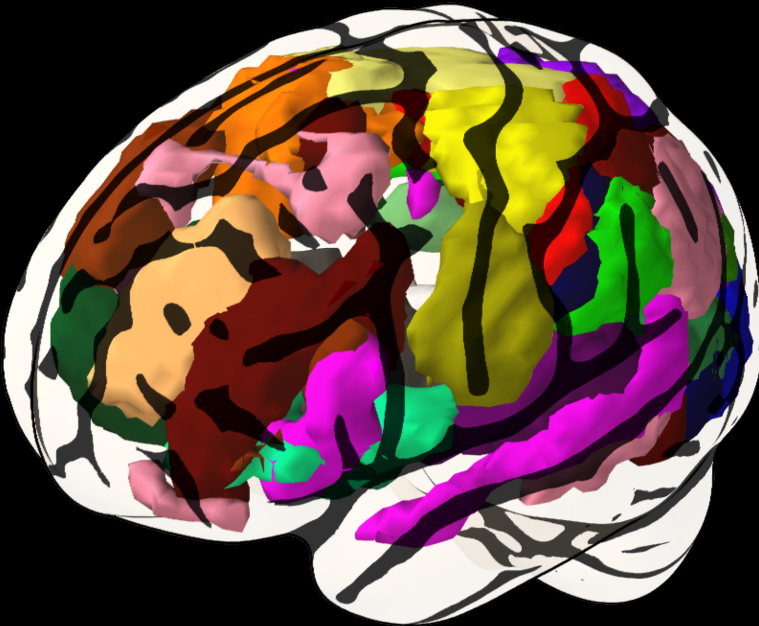






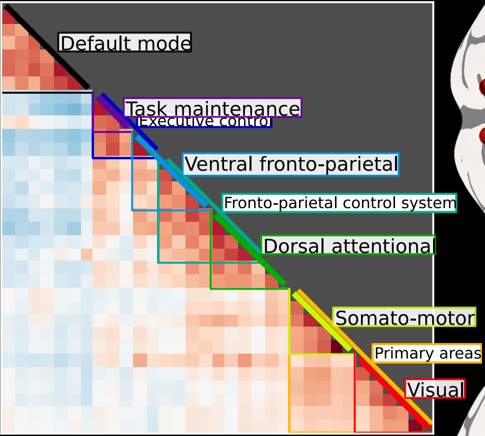


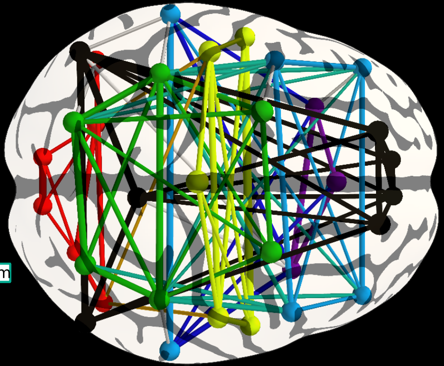
2 A full-brain parcellation

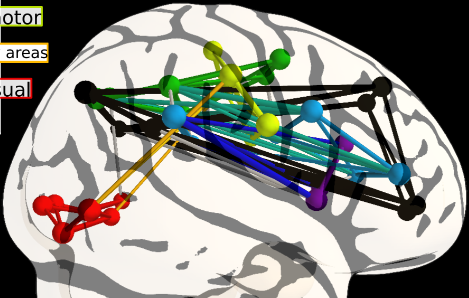


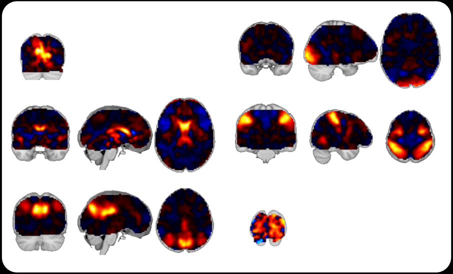
2 Between-regions connectivity

Correlation matrix Σ

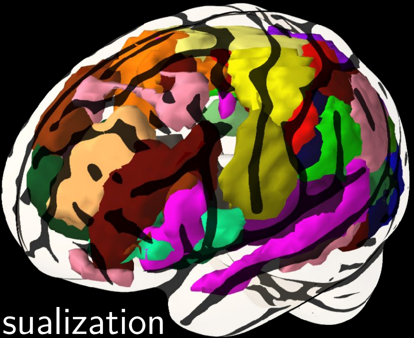








Data



Visualization

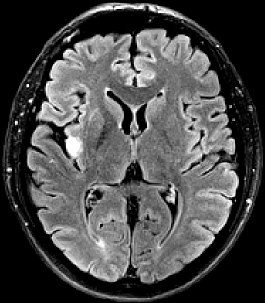
Change of representation

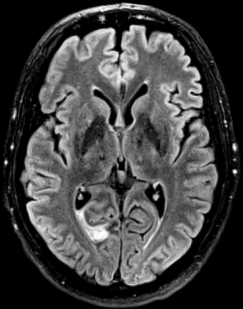
Understanding complex data requires interactive visualization with ***high level concepts***

Mayavi:

Python 3D visualization

3 Inter-subject comparisons

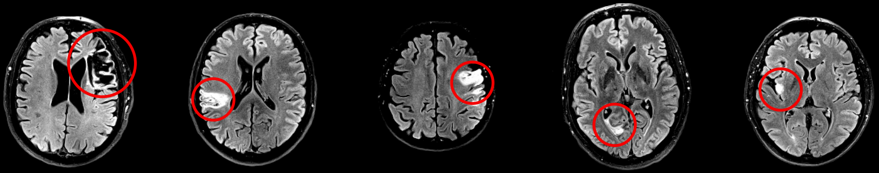




Ischemic stroke:

- Temporary interruption of blood flow
- Affects 1 person out of 100 every year for people $>$ 55 years
- Causes focal lesions of varying consequences

motor deficiencies   language impairments   coma   ...



How does brain reorganize after stroke?

Prognostic based on intrinsic brain activity?

Probabilistic model of data

- Covariance = 2^{nd} moment of observed data

$\Rightarrow$ Specifies a probability distribution

Test the likelihood of data in a covariance model

3 Probabilistic covariance modeling

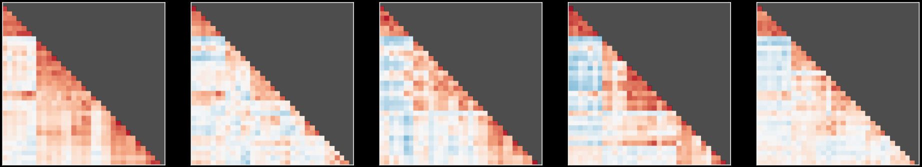
Probabilistic model of data

- Covariance = 2[nd] moment of observed data

⇒ Specifies a probability distribution

Test the likelihood of data in a covariance model

Covariances variations in healthy population



Which one of the above has a large cortical lesion?

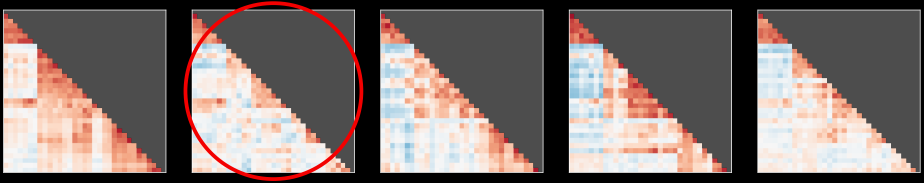
Probabilistic model of data

- Covariance = 2$^{\text{nd}}$ moment of observed data

⇒ Specifies a probability distribution

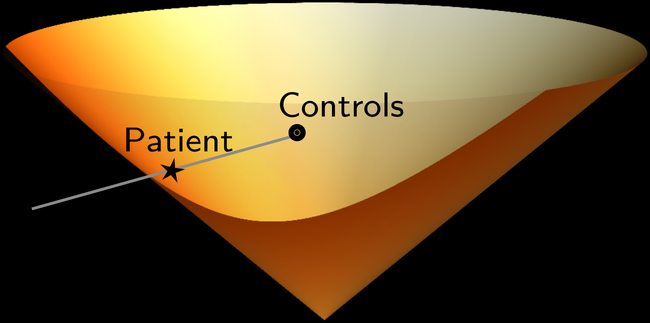
Test the likelihood of data in a covariance model

Covariances variations in healthy population

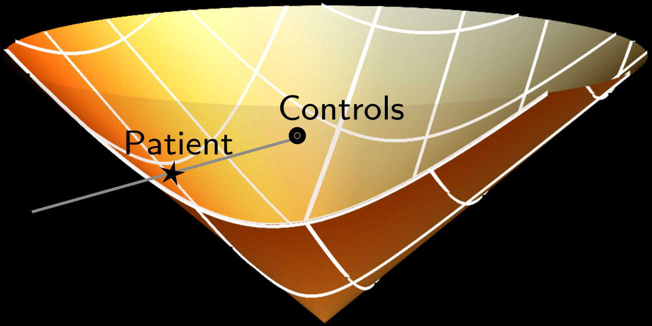


Which one of the above has a large cortical lesion?

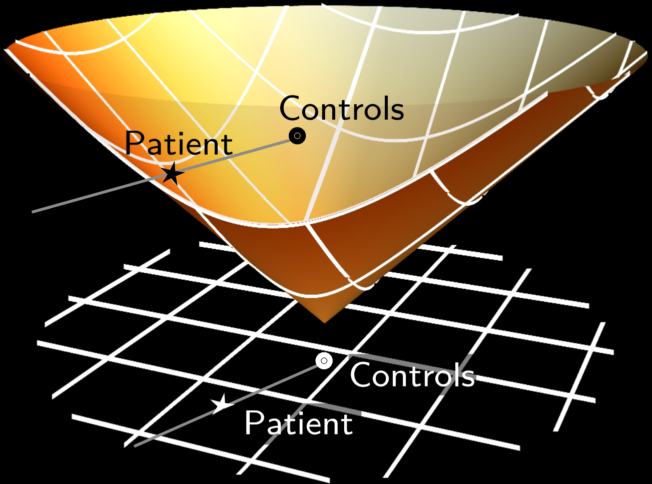
3 Modeling variability of covariance



3 Modeling variability of covariance

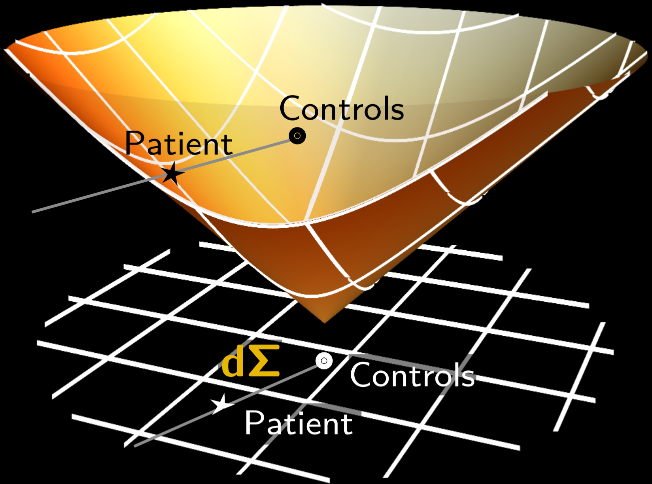


3 Modeling variability of covariance



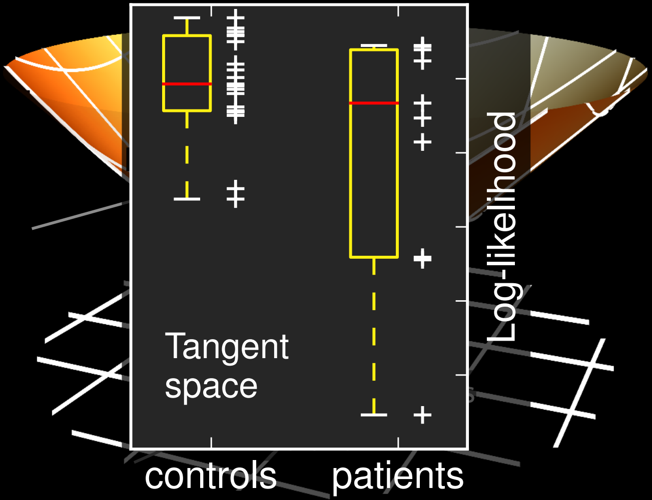
Varoquaux, MICCAI 2010

3 Modeling variability of covariance



$\mathcal{P}(\mathbf{d\Sigma})$: probability density in tangent space

3 Modeling variability of covariance



$\mathcal{P}(\mathbf{d\Sigma})$: probability density in tangent space

Between which regions is connectivity is modified?

Ill-posed problem

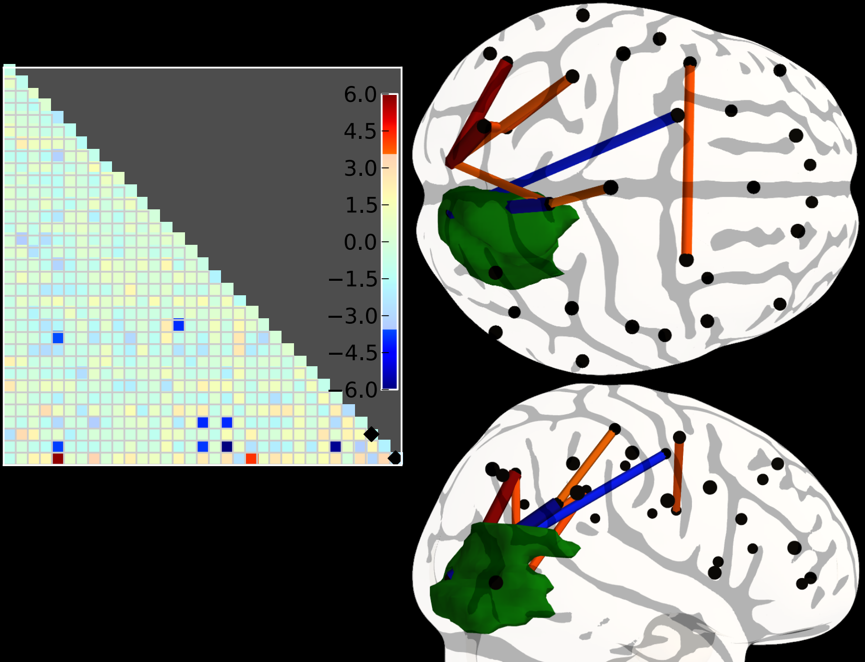
- Non-local effects

$\Rightarrow$ Many differences causes give the same observations

Our suggestion

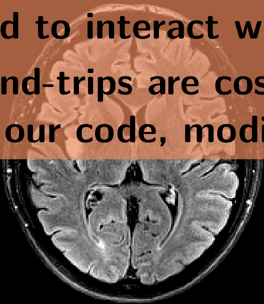
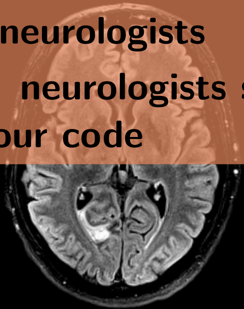
- Pair-wise partial correlations
- In tangent space: almost independent
- Draw random groups of healthy controls to tabulate their variability

3 Finding the cause of the difference



Research code in clinical settings

- Applications give rise to non-trivial mathematical problems
- Need to interact with neurologists
- Round-trips are costly: neurologists should use our code, modify our code

4 From models to software tools?

4 The hidden costs of releasing software

- Gap from paper to software:
  Remove duplication   Write documentation
  Make usable APIs   Write tests   Fix corner cases

Cost of code

Complexity scales as the square of project size

Woodfield 1979, *an experiment on unit increase in problem complexity*

Cost of users

- Backward compatibility
- Support for multiple installations and versions
- Bug reports, feature request, mailing list support

Maintenance cost $\sim (\#\ \text{lines})^2 \sqrt{\#\ \text{users}}$

4 Addressing the scientific software challenge

Better code

- High-level coding and abstractions
  - numpy arrays: abstract out memory and pointers
  - traits Model+View: hide dialogs and events
  - joblib: factor out persistence
- Common libraries
  scipy, Mayavi, ...

Project management decisions

- 80/20 rule
- Not every research code should be released
- Focus on documentation and installation

4 Software as building blocks for new science

Segregated, functionally-specialized, packages

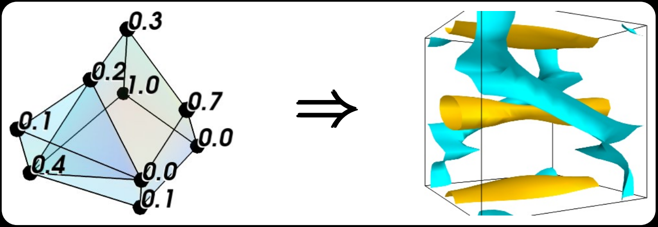
- Answer a specific problem
- Limit dependencies

Reusable projects

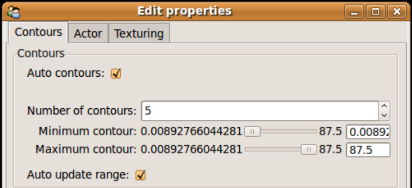
- Useful for a different purpose than the original one
- Libraries (no control of point of entry)
- Standard data structures
- Most often simple
- BSD licensed

4 Mayavi: making 3D visualization reusable

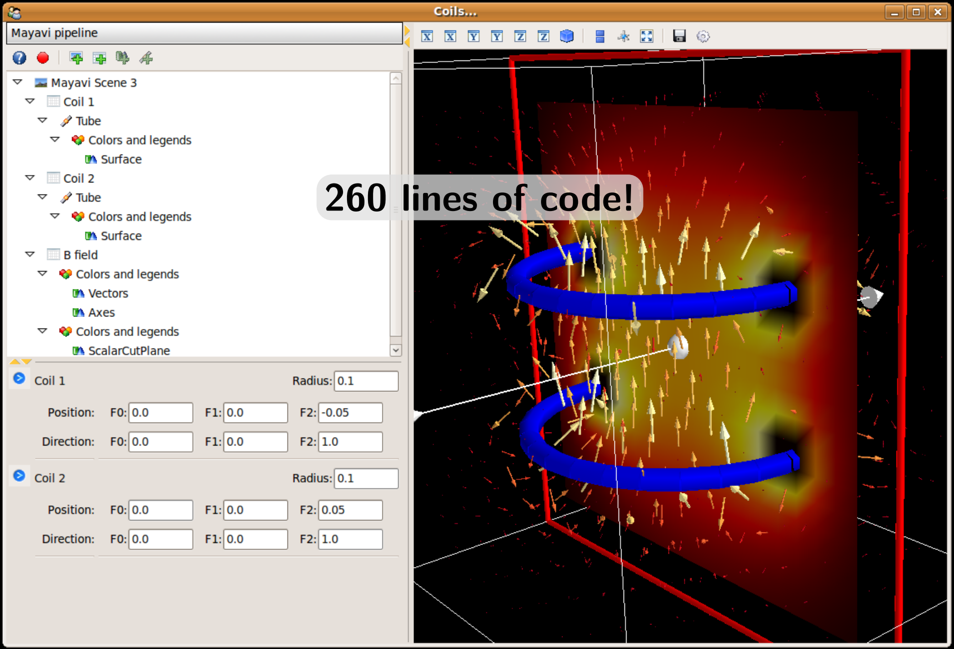
Pipelines: from data sources to visualization objects



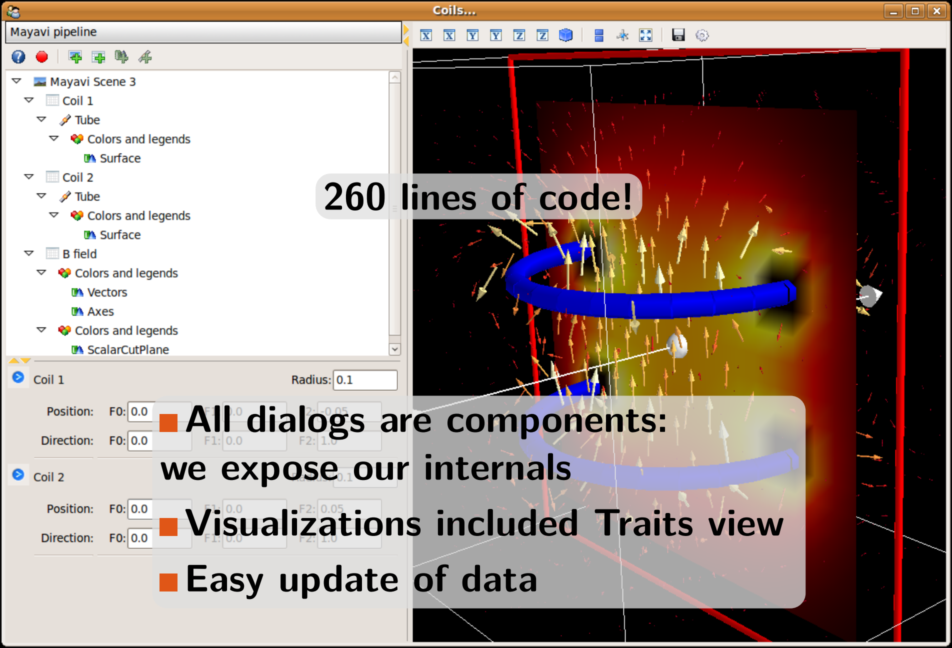
- Simple API: `mlab.contour3d(x, y, z, data)`
- Building pipelines by function calls:
  mlab.pipeline.iso_surface(mlab.pipeline.contour(src))
- GUI
  + automatic script generation



4 Mayavi: making 3D visualization reusable



4 Mayavi: making 3D visualization reusable



260 lines of code!

- All dialogs are components: we expose our internals
- Visualizations included Traits view
- Easy update of data

4 joblib: not writing pipelines

Dataflow pipeline: *succession of processing steps executed on demand*

---

joblib:

- Lazy-revaluation
- Persitence
- Parallel processing
- Logging

All with functions (seemingly)

4 joblib: not writing pipelines

```
>>> from joblib import Memory
>>> mem = Memory(cachedir='/tmp/joblib')
>>> import numpy as np
>>> a = np.vander(np.arange(3))
>>> square = mem.cache(np.square)
>>> b = square(a)
________________________________________________________________________________
[Memory] Calling square...
square(array([[0, 0, 1],
       [1, 1, 1],
       [4, 2, 1]]))
___________________________________________________________square - 0.0s, 0.0min

>>> c = square(a)
>>> # The above call did not trigger an evaluation
```

Towards Quantitative modeling of spontaneous brain activity

- Requires probabilistic models and state-the-art machine learning tools
- Algorithms and software development hand in hand with neurologists for applications
- Need a high-level stack of software tools general purpose with separation of concerns