

Neuroimaging Analysis Kit - Overview

Pierre Bellec

pierre.bellec@criugm.qc.ca

Feindel Brain Imaging Lecture - BIC/MNI/McGill 2016/10

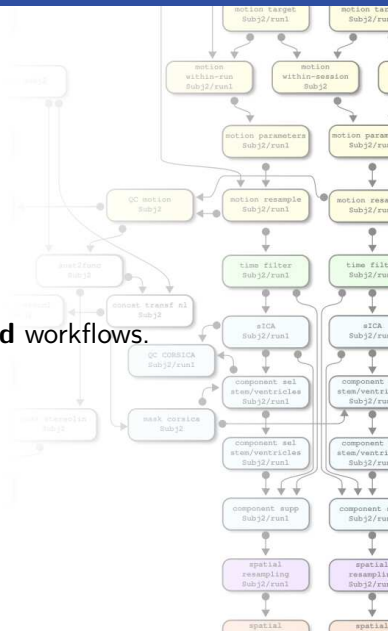
Département d'informatique et de recherche opérationnelle, Université de Montréal



What's NIAK

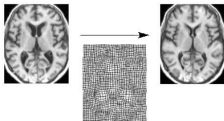
The Neuroimaging Analysis Kit (NIAK):
a software package for connectivity analysis
in large fMRI datasets.

- ▶ A catalogue of **complete workflows**.
- ▶ **Scales** for large datasets / analyses.
- ▶ **Reproducible** deployment of **well tested workflows**.
- ▶ Web-based **notebook** interface.
- ▶ Interactive **dashboard** reports.
- ▶ Free and **open-source** (MIT).



Available pipelines

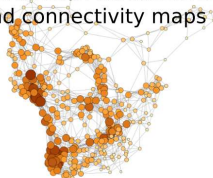
fMRI preprocessing



Functional brain parcellation



Generation of connectomes and connectivity maps

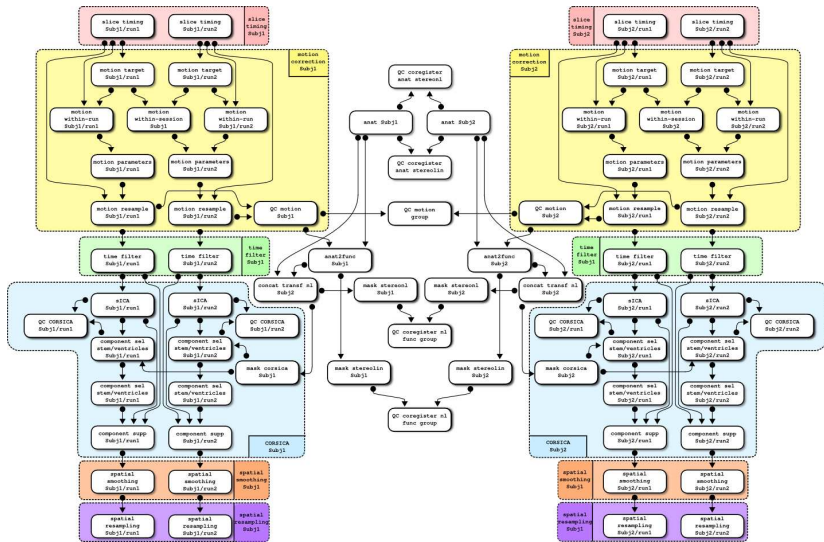


Check <http://niak.simexp-lab.org> for more info.

Main dependencies

- ▶ **Ubuntu:** An operating system based on mostly GNU tools as well as the linux kernel. Free software (mixed licenses).
<https://www.ubuntu.com/>
- ▶ **Octave:** A high-level scientific programming language, largely identical to Matlab. Octave is Free Software (GNU license).
<https://www.gnu.org/software/octave/>
- ▶ **The MINC toolkit:** A set of command line tools for brain registration, segmentation and basic image processing operation. Underlying code is mostly C and PERL. Free software (MIT like custom license). <http://bic-mni.github.io/>
- ▶ **The brain connectivity toolbox:** A toolbox to generate properties of brain networks. <https://sites.google.com/site/bctnet/>.

One pipeline, many jobs...



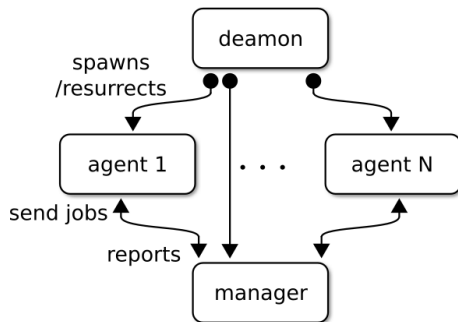
Jobs and dependencies for fMRI preprocessing of two subjects with two functional runs each.

The pipeline system for Octave and Matlab

NIAK is powered by PSOM, an open-source library for scripting pipelines using Octave or Matlab (Bellec et al., Frontiers in Neuroinformatics, 2012).

- ▶ **Parallel computing:** Detection and execution of parallel components in the pipeline. The same code can run in a variety of execution environments (local, multi-core, cluster).
- ▶ **Provenance tracking:** Generation of a comprehensive record of the pipeline stages and the history of execution.
- ▶ **Fault tolerance:** Multiple attempts will be made to run each job before it is considered as failed. Failed jobs can be automatically re-started.
- ▶ **Smart updates:** When an analysis is started multiple times, only the parts of the pipeline that need to be reprocessed are executed.

<http://psom.simexp-lab.org>

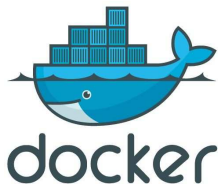


PSOM features an agent-based execution model.

Benchmark PSOM 2.0

- ▶ Dataset Human Connectome Project, 875 subjects with T1 + 7 multiband fMRI task runs.
- ▶ 123k jobs / 3.4 T raw input / 3.8 T output / 173k unique input/output files.
- ▶ guillimin: supercomputer (Xeon, 20k+ cores on 2016), infiniband parallel file system.
- ▶ Up to 300 concurrent processes allowed.
- ▶ Serial time: 17.9k hours / 746.87 days. Parallel time: 70 hrs. Parallelization efficiency: 85%
- ▶ deviation from 100% efficiency mostly attributable to queuing delays in order to access resources.


NIAK deployment using Docker














- ▶ The NIAK now as a docker container, available in docker hub <https://hub.docker.com/>, as well as singularity <http://singularity.lbl.gov/>, designed for high-performance computing infrastructures.
- ▶ The container includes all dependencies (MINC-toolkit, Octave, PSOM, NIAK, Brain Connectivity Toolbox, Jupyter).
- ▶ This facilitates installation and increases reproducibility on all platforms, Linux, Mac, Windows
http://niak.simexp-lab.org/niak_installation.html

How does it work?

Octave (similar to matlab) runs in a jupyter notebook.

 **jupyter** niak_tutorial_fmri_preprocessing Last Checkpoint: 4 hours ago (autosaved)

File Edit View Insert Cell Kernel Help Octave 

         Markdown  CellToolbar

fMRI preprocessing

This tutorial shows how to run the NIAK fMRI preprocessing pipeline, using a limited set of options. See the [documentation](#) of the pipeline for a more comprehensive list of options. Download the tutorial as a notebook [here](#) and a matlab script [here](#). We recommend to use [jupyter](#) from a niak docker container, as described in the [NIAK installation page](#).

Preparing files

First download a small fMRI dataset, with a structural scan. Be aware that all raw and derivatives data will be generated in the current folder. Note that you will need to manually remove the `data_test_niak_mnc1` and `fmri_preprocess` folders to restart this tutorial from scratch.

```
In [1]: clear
niak_wget('data_test_niak_mnc1');

--2016-10-15 18:36:07-- http://www.nitrc.org/frs/download.php/7241/data_test_niak_mnc1.zip
Resolving www.nitrc.org (www.nitrc.org)... 132.239.16.23
```

Testing pipelines...



Numerical instabilities creep up in a complex pipeline.

Effective numerical stabilization strategies are required to extract reliable measures.

An engineering problem surprisingly little studied in the fMRI field.



Continuous integration tests

NIAK continuous integration tests running on
<https://circleci.com/gh/SIMEXP/niak>

The screenshot displays the CircleCI interface for a build. At the top, the breadcrumb navigation shows 'Builds » SIMEXP » niak » master » build 89'. To the right are buttons for 'Rebuild' and 'Project Settings'. The build status is 'SUCCESS', with a green checkmark icon. Below this, the build details are listed: 'Finished: 2 days ago (25:03)', 'Previous: 88', 'Parallelism: 1x out of 4x', and 'Queued: 00:48 waiting + 00:00 in queue'. The 'Triggered by' section shows 'P-O Quirion (pushed 671a0ab)'. A section titled 'COMMITTS (1)' lists the commit 'Pierre-Olivier Quirion 671a0ab can run jupyter non root'. Below this is a table with tabs for 'Test Summary', 'Queue (00:48)', 'Debug via SSH', 'Artifacts', 'circle.yml', and 'Build Timing'. The 'Test Summary' tab is active, showing a tree view of the test results. The tree structure is as follows:

- Container 0
 - home/
 - ubuntu/
 - niak/
 - result/
 - demoniak_connectome/
 - demoniak_glm_connectome/
 - demoniak_glm_fir/
 - demoniak_region_growing/
 - demoniak_scores/
 - demoniak_stability_fir/
 - demoniak_stability_rest/
 - glm_connectome_unit/
 - logs/

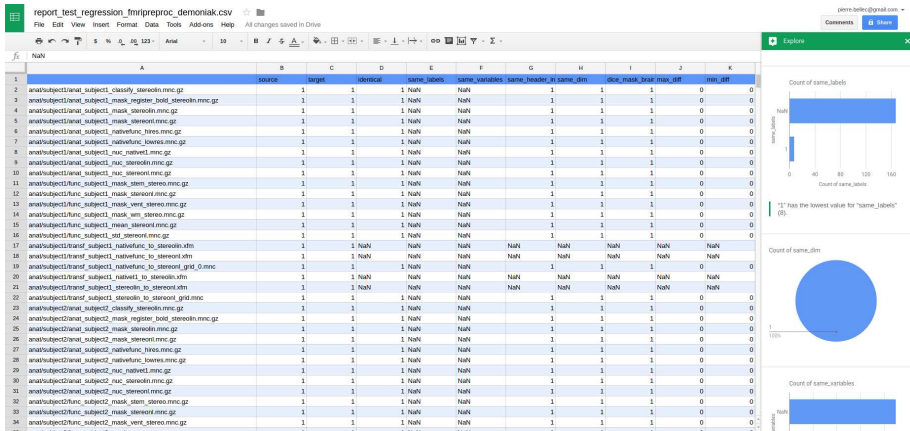
Below the tree view, the following files are listed:

 - report_test_regression_connectome_demoniak.csv
 - report_test_regression_fmripredproc_demoniak.csv
 - report_test_regression_glm_connectome_demoniak.csv

A question mark icon is visible in the bottom right corner of the interface.

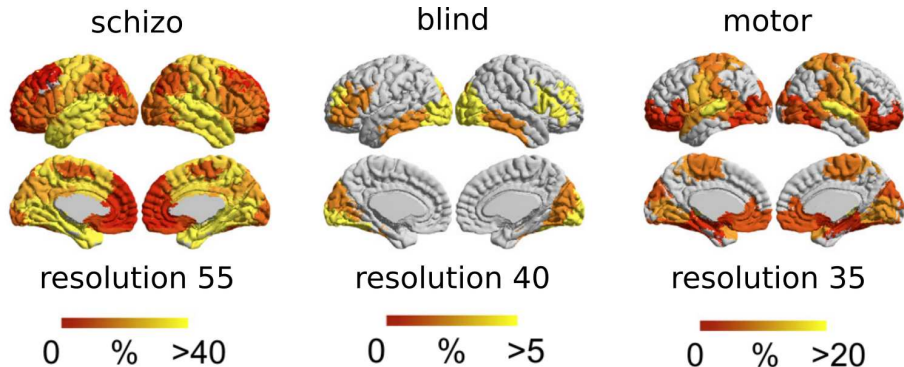
Continuous integration tests

Each change in NIAK triggers a comparison between current results and a fixed, target version, across all available pipelines. Quantitative reports show which stage of the pipeline has changed, and by how much.



Large-scale validation at release

Future NIAK releases will systematically replicate a number of key large-scale validation experiments and compare results across versions.



Between-group comparisons in resting-state connectivity across three populations. See Bellec et al., Orban, Neuroimage 2015.

Acknowledgements

- ▶ **Funding:** Brain Canada CBRAIN, UdM, UNF/CRIUGM, CCNA (CIHR), Courtois foundation, Lemaire foundation.
- ▶ **QC@NIAK:** **Yassine Benhajali**, Sebastian Urchs, AmanPreet Badhwar, Perrine Ferré, Angela Tam, Christian Dansereau.
- ▶ **validation@NIAK:** **Pierre Orban**, Yassine Benhajali, Felix Carbonell, Christian Dansereau, Geneviève Albouy, Maxime Pelland, Cameron Craddock, Olivier Collignon, Julien Doyon, Emmanuel Stip.
- ▶ **NIAK:** Pierre-Olivier Quirion, Angela Tam, Sebastian Urchs, Yassine Benhajali, Christian Dansereau, Felix Carbonell, Jussi Tohka. Many indirect contributions

The NIAK project is under an MIT opensource license

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.