



**NIGHRES**

Neuroimaging at  
high resolution

Warning:  
May contain subjective  
views

# The Nighres Toolbox

Pierre-Louis Bazin

[pilou.bazin@uva.nl](mailto:pilou.bazin@uva.nl)

Integrative Model-based Cognitive Neuroscience research unit  
Universiteit van Amsterdam



Integrative Model-based Cognitive Neuroscience  
Research Unit



# High resolution MRI at 7T

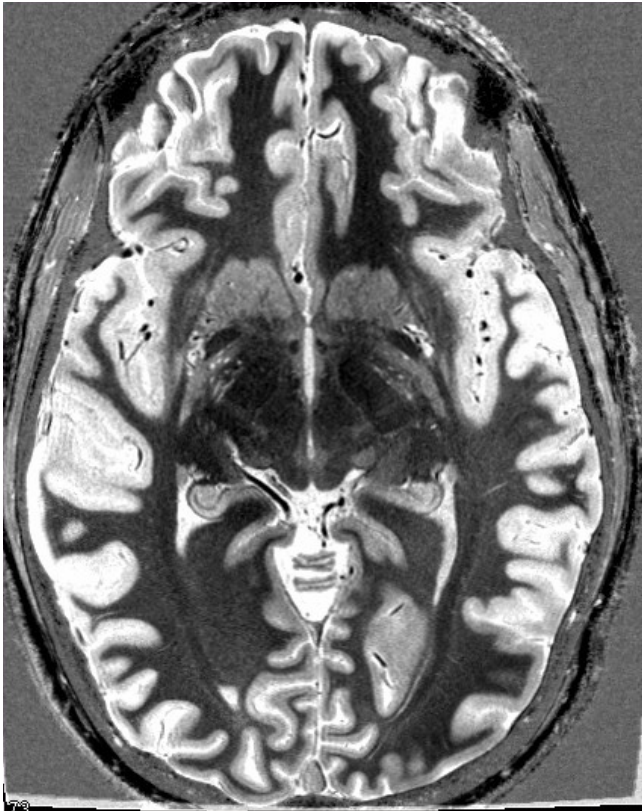
Increased anatomical resolution



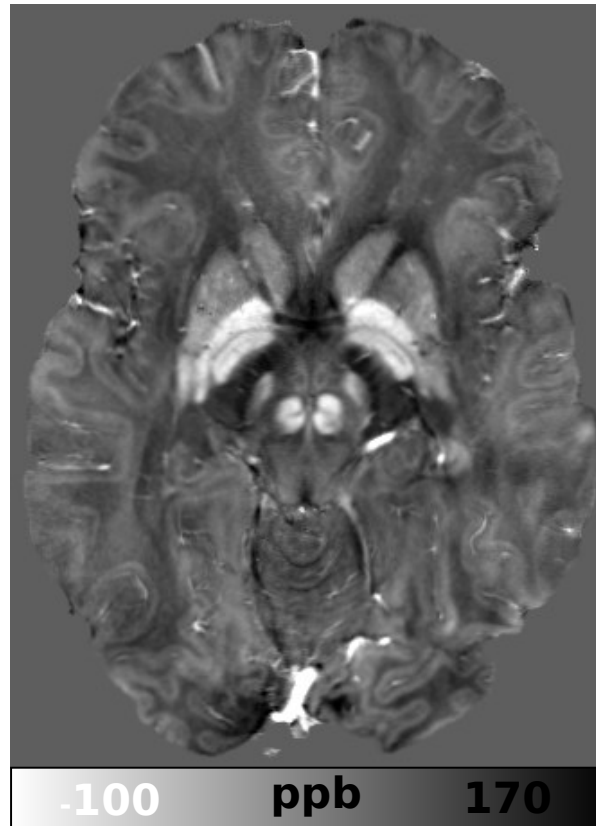
*[Gallichan and Marques, ESMRMB 2015]*

# High resolution MRI at 7T

Increased anatomical resolution



Quantitative contrasts

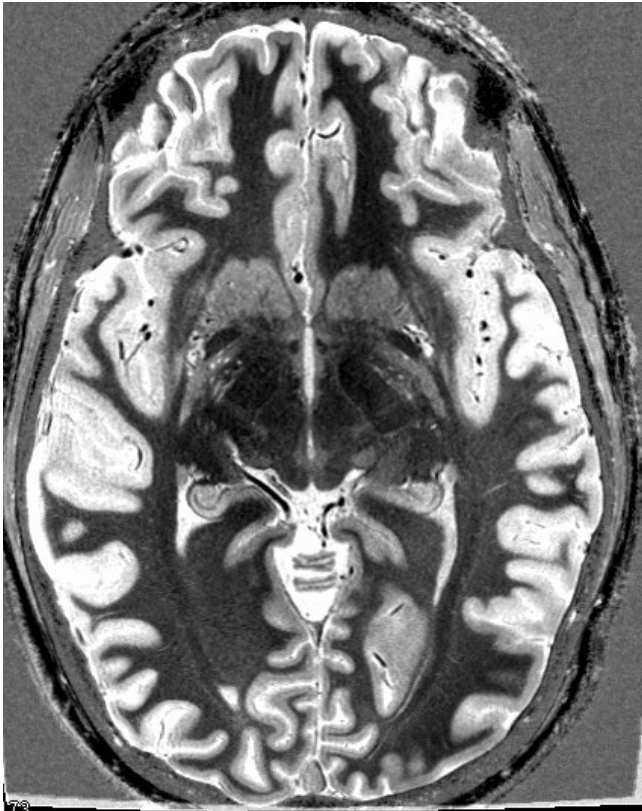


*[Gallichan and Marques, ESMRMB 2015]*

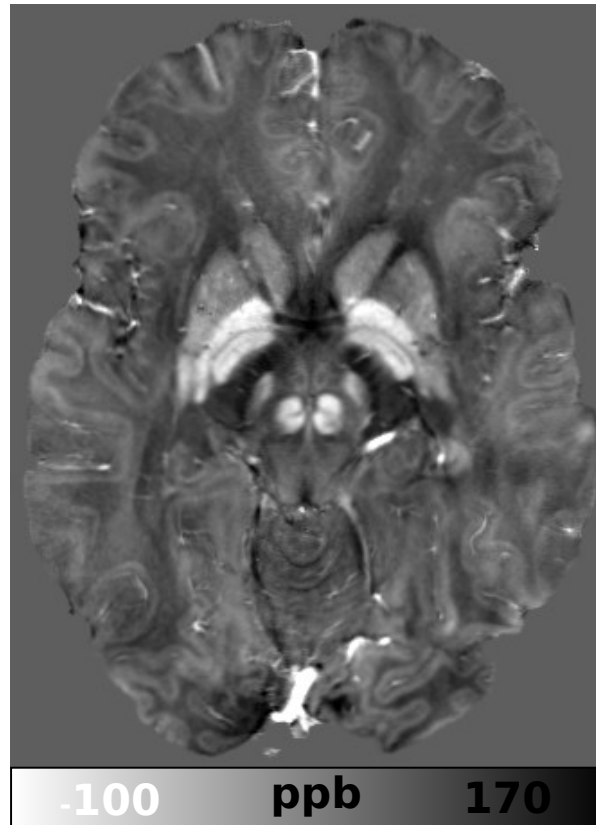
*[Deistung A et al. Neuroimage. 2013]*

# High resolution MRI at 7T

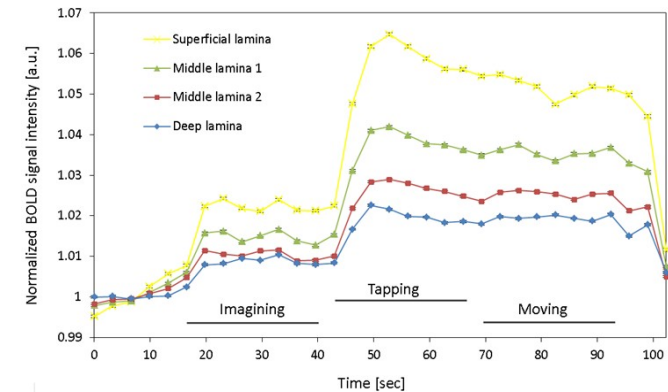
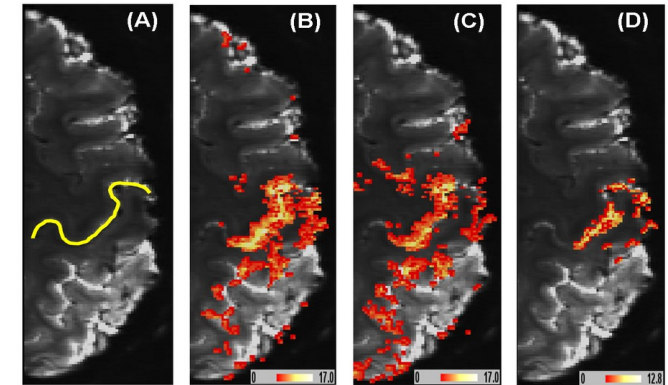
Increased anatomical resolution



Quantitative contrasts



Intra-cortical fMRI



[Gallichan and Marques, ESMRMB 2015]

[Deistung A et al. Neuroimage. 2013]

[Trampel et al., SfN 2011]



# High resolution MRI challenges at 7T

Increased anatomical resolution



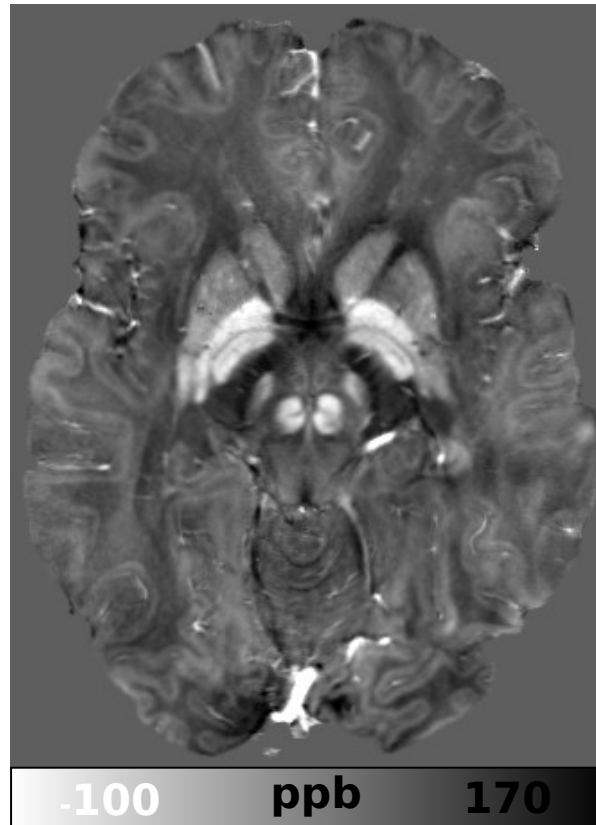
Explosion of data size and processing times

# High resolution MRI challenges at 7T

Increased anatomical resolution



Quantitative contrasts



Explosion of data size and processing times

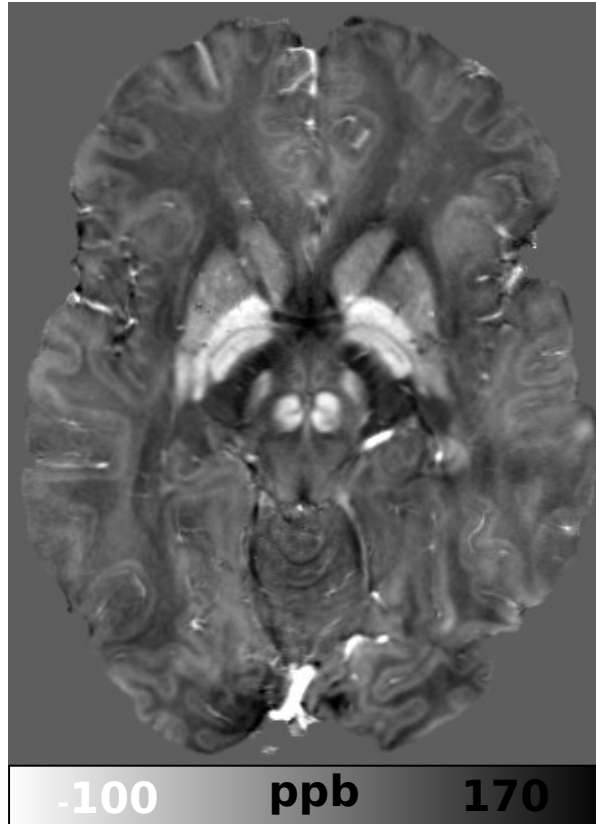
New models of the signal needed

# High resolution MRI challenges at 7T

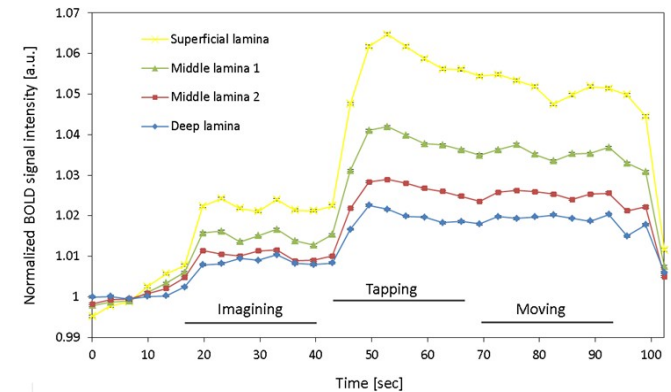
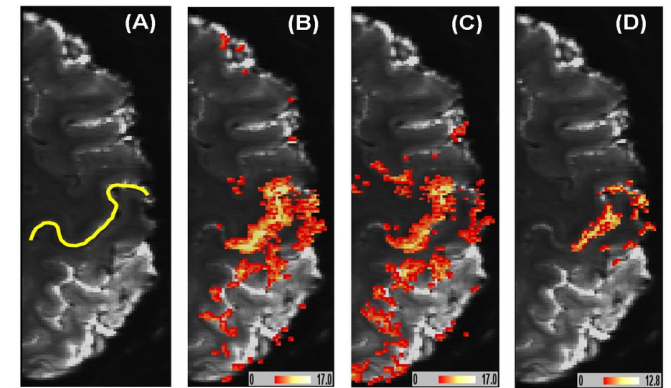
Increased anatomical resolution



Quantitative contrasts



Intra-cortical fMRI



Explosion of data size and processing times

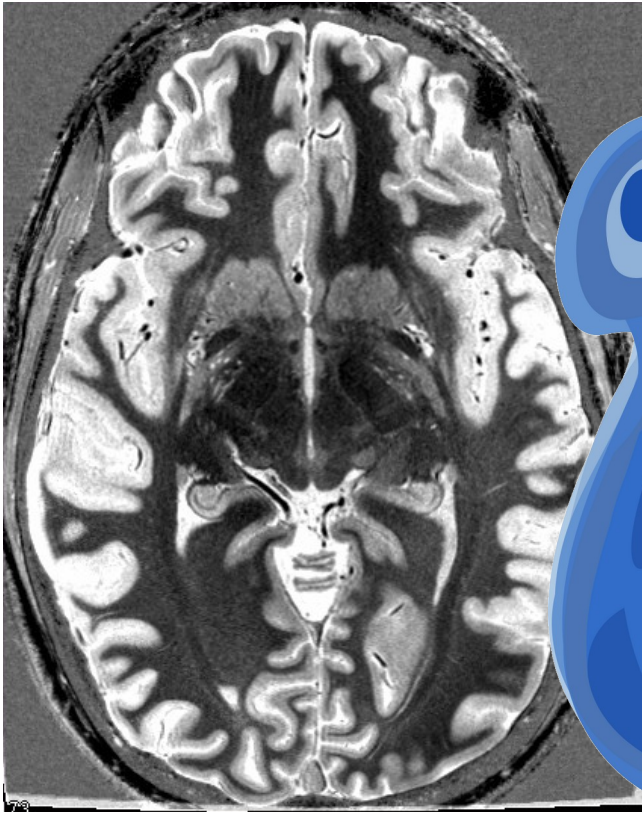
New models of the signal needed

Increase in required spatial precision

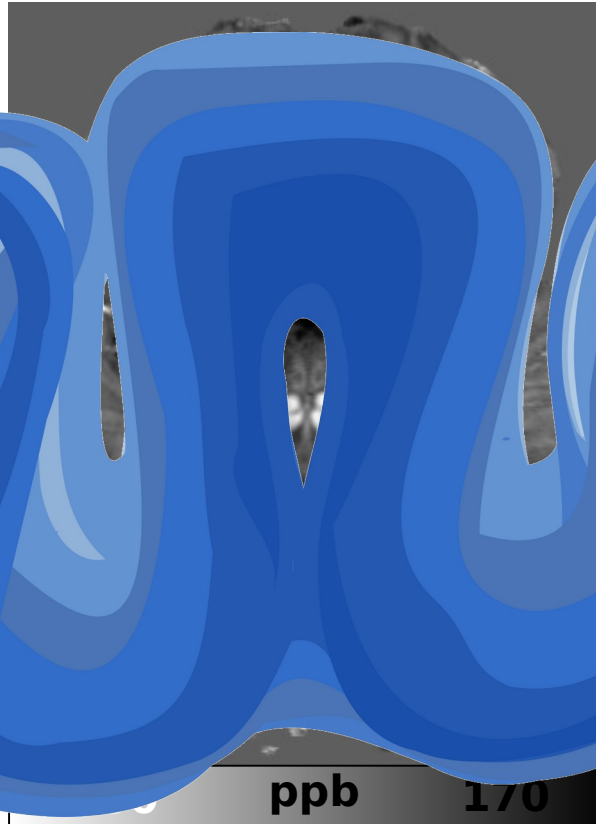


# High resolution MRI challenges at 7T

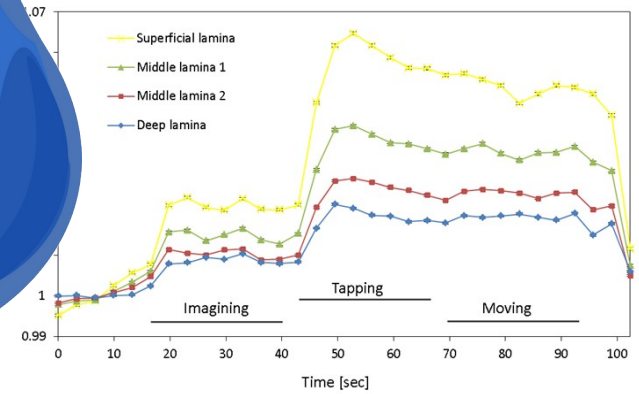
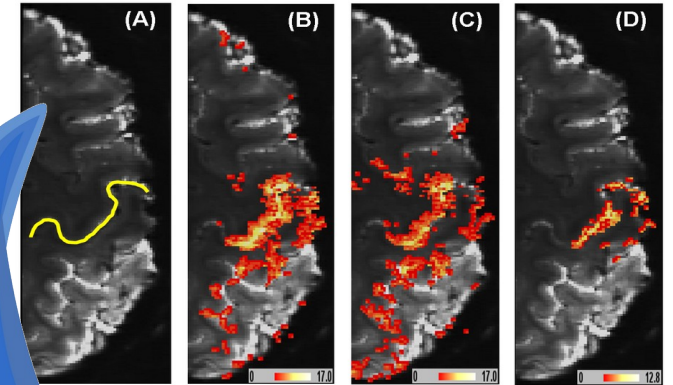
Increased anatomical resolution



Quantitative contrasts



Intra-cortical fMRI



Explosion of data size and processing times

Optimized numerical methods

New models of the signal needed

Quantitative MRI reconstruction

Anatomy-driven approaches

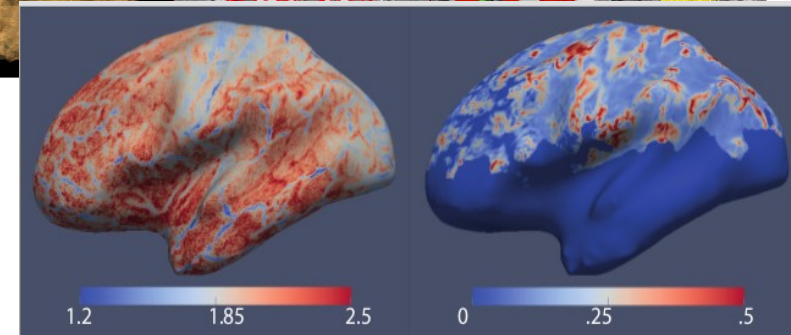
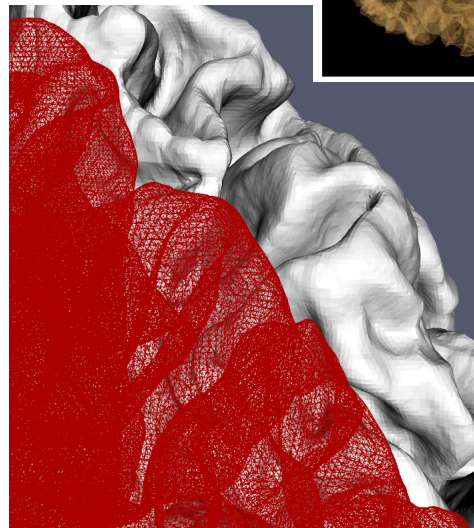
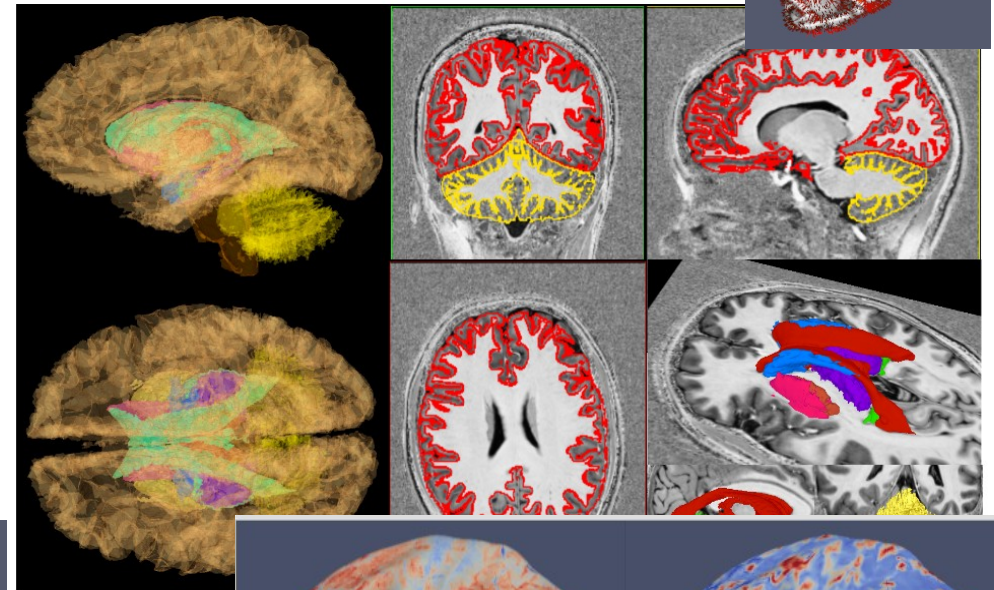
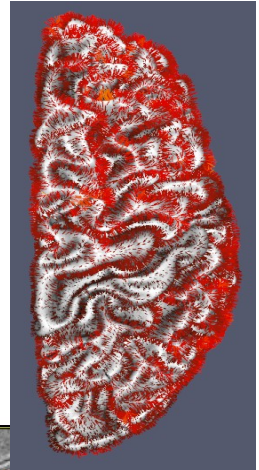
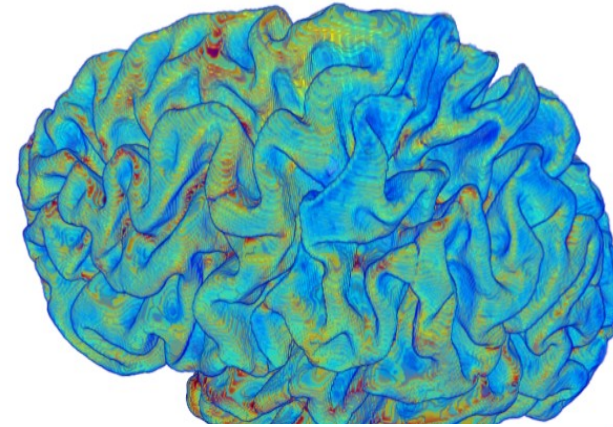
Increase in required spatial precision



# What is Nighres?

High resolution image processing algorithms **designed for 7T data**

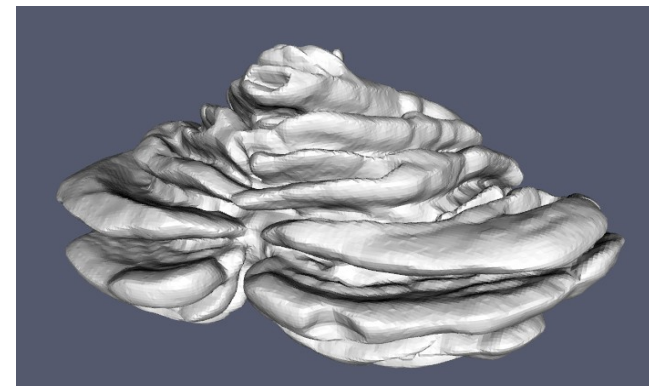
- Handles MP2RAGE data, other **quantitative contrasts**
- **Segmentation**: cortical and sub-cortical structures
- Cortical **surface** extraction and inflation (cerebral and cerebellar)
- Processing in MNI space at **0.4 mm**
- Highly accurate cortical **layering, profiling** and **co-registration**



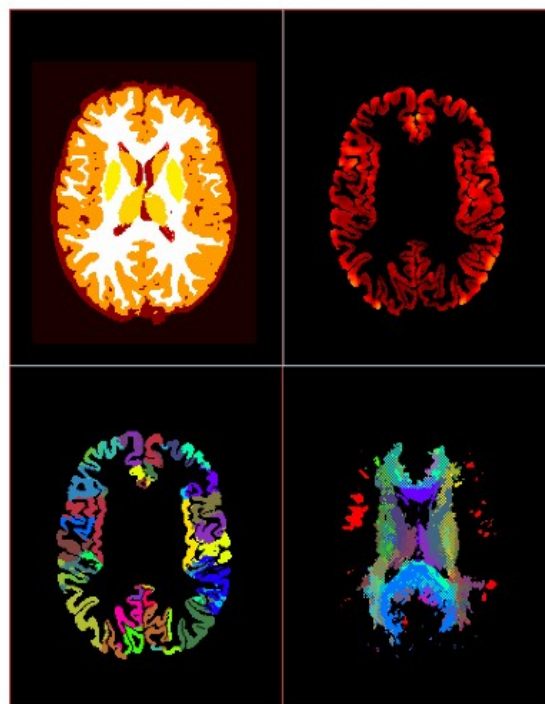
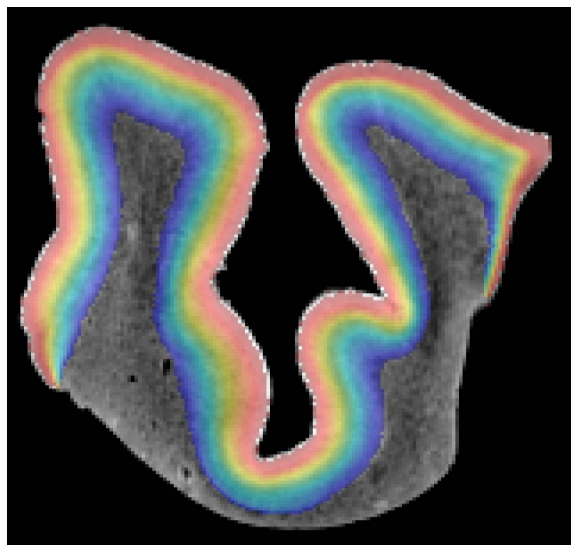
[Bazin et al., 2014]  
[Tardif et al., 2015]  
[Waehnert et al., 2016]

# Can Nighres handle 3T data?

- Many prior tools developed for 3T data included (see Landman et al. Neuroinformatics 2012)
- Most 7T processing tools are 3T compatible
- Easy interfacing with python-based tools

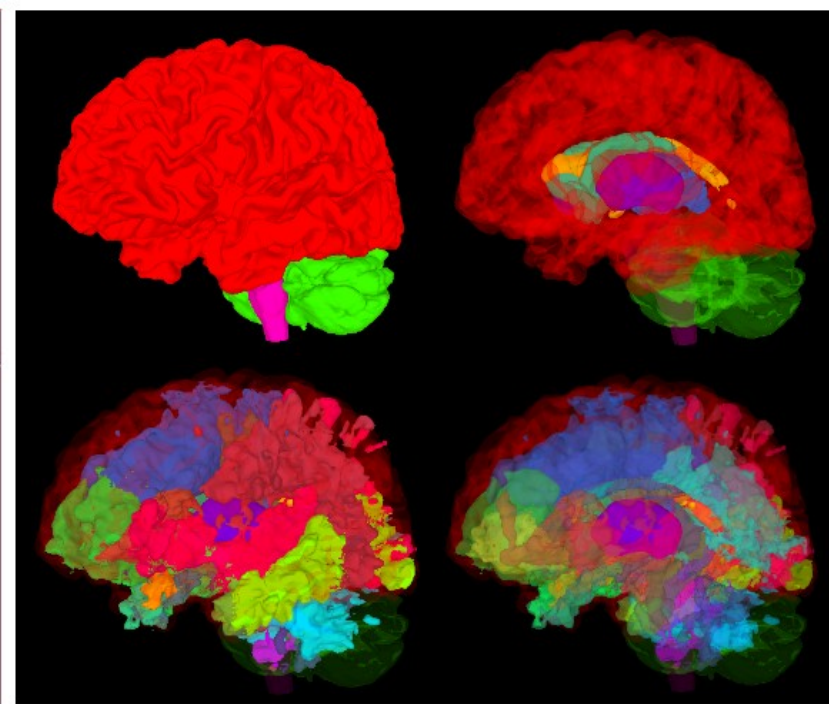


Main structures    Cortical thickness



Cortical gyri

WM tracts

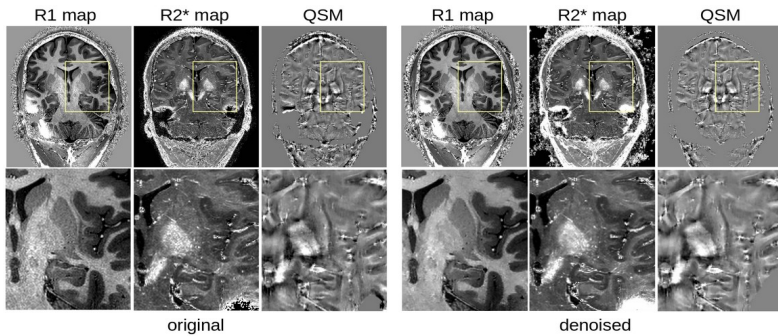


3D renderings with various levels of transparency

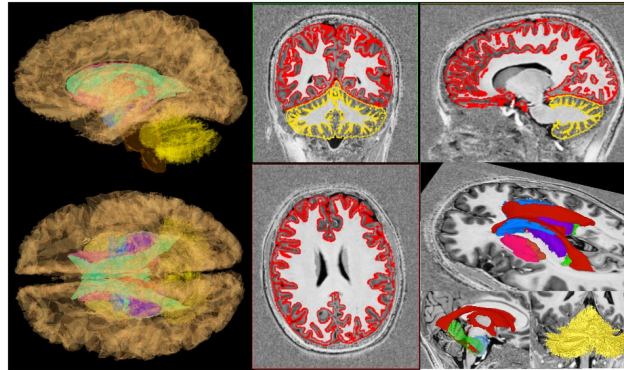


# A growing collection of tools

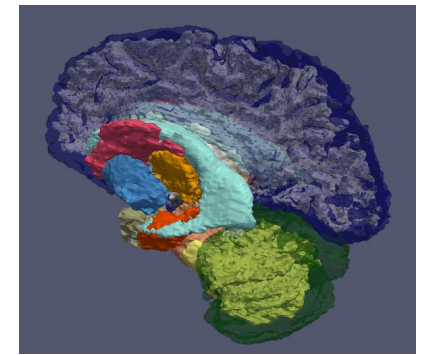
### Quantitative MRI reconstruction and denoising



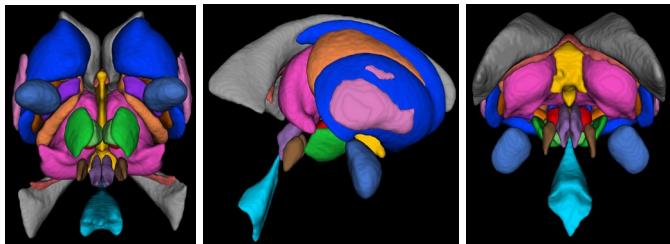
### MP2RAGE Whole brain segmentation



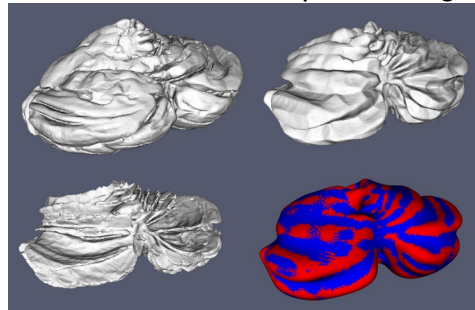
### DWI segmentation



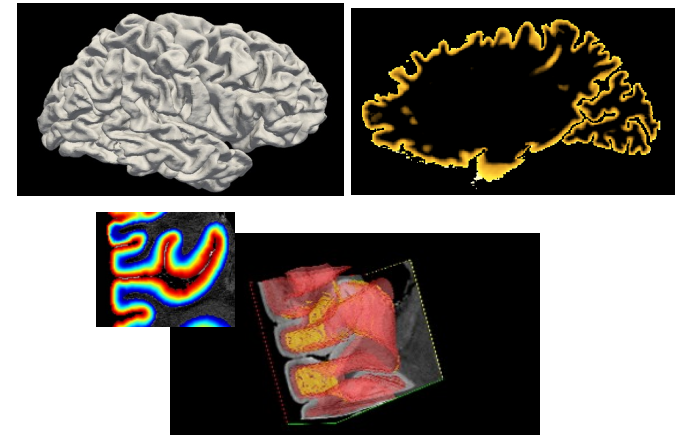
### Subcortical Parcellation



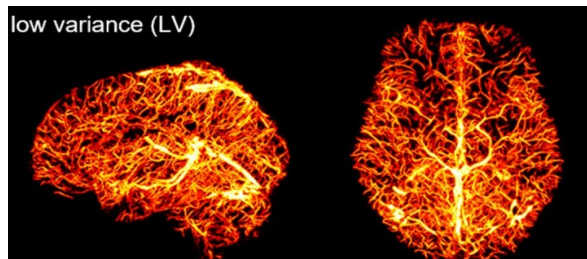
### Cerebellar surface processing



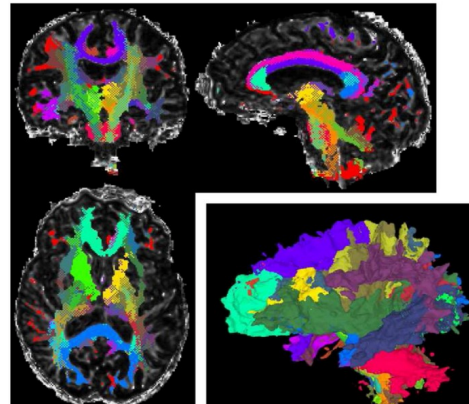
### Cortical & laminar analysis



### Vascular reconstruction



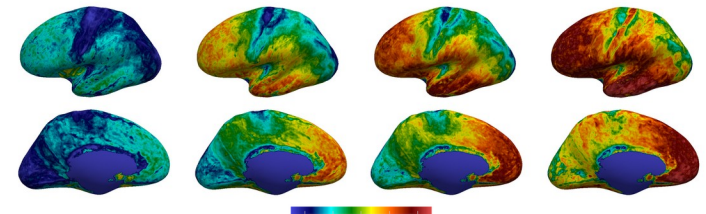
### White matter tract labelling



### Surface-based registration



### Volume-preserving lamination



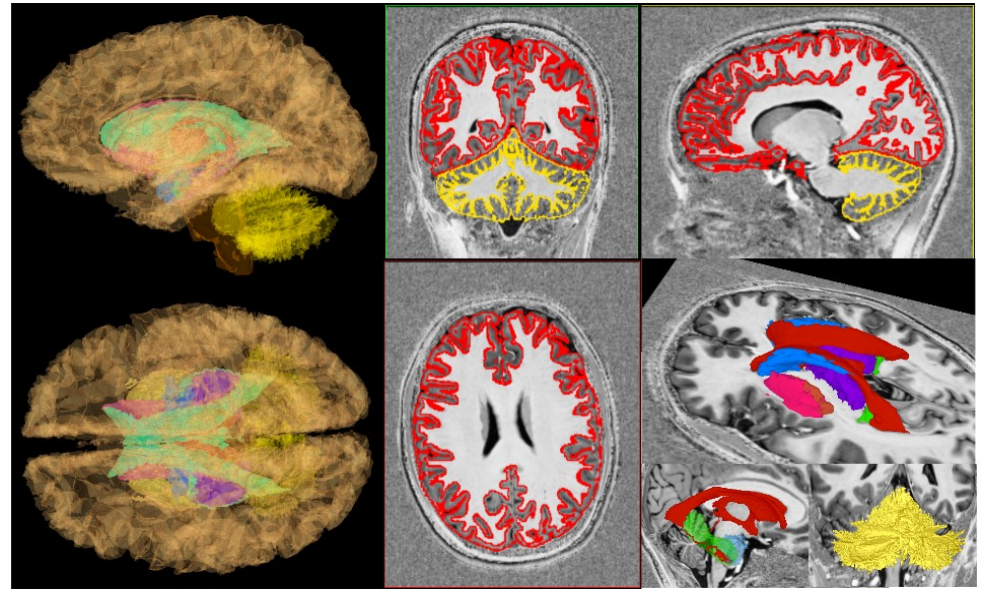
# Brain segmentation

## *Handled base contrasts:*

- Quantitative T1 at 3T, 7T, 9.4T
- HCP T1w, T2w data
- MPRAGE 3T (ADNI)
- MPM at 3T
- DWI 3T (FA and MD)
- R2\* and QSM
- Further extensible...

## *Segmented structures:*

cerebral & cerebellar cortices,  
sub-cortical nuclei,  
vasculature,  
white matter tracts...



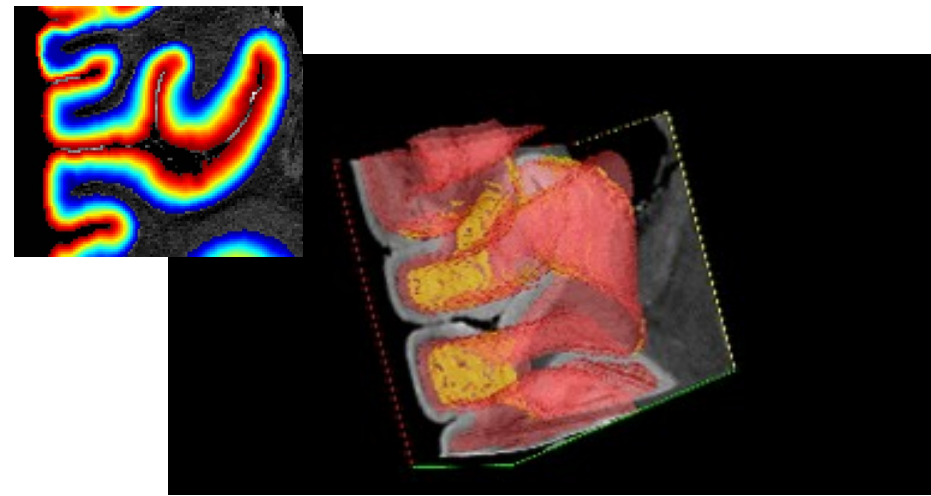
## *Additional tools & filters:*

CSF partial voluming,  
intensity normalization,  
skull stripping,  
background suppression,  
exponential fitting,  
noise estimation and regularization,  
...

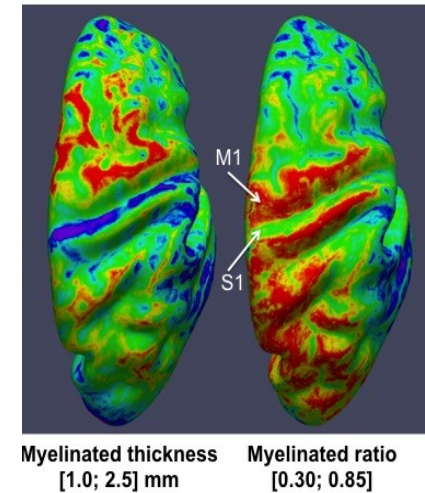
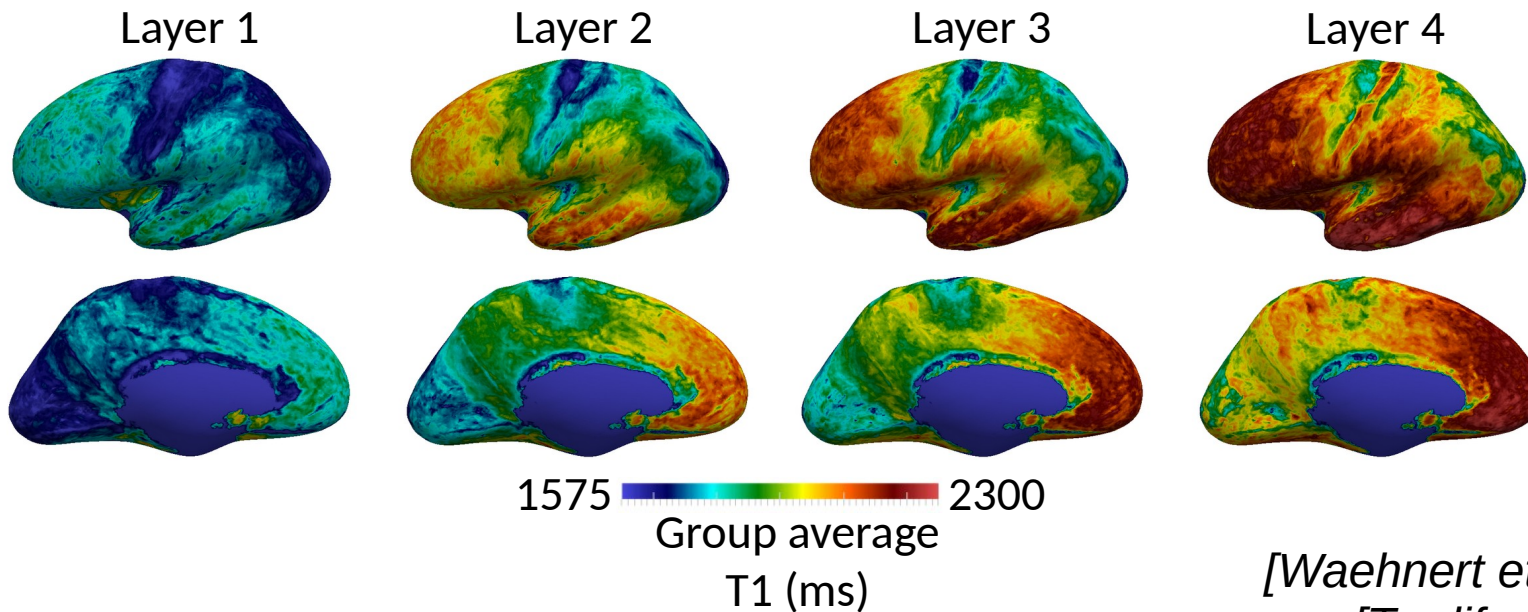


# Cortical analysis

- Cortical reconstruction and surface inflation with CRUISE
- Anatomical representation of depth
- Multi-contrast alignment
- Mixed contrast and morphology measures



Volume-preserving lamination

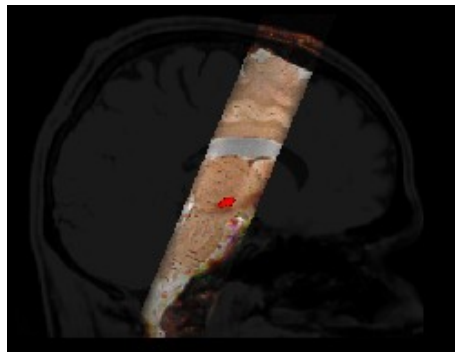
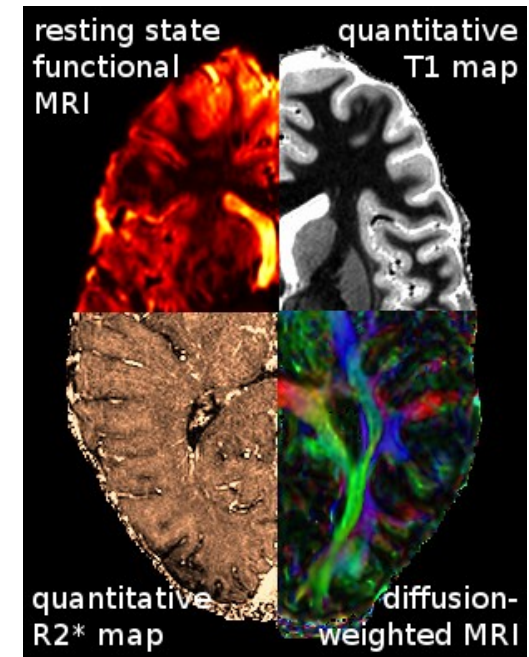


Myelinated thickness

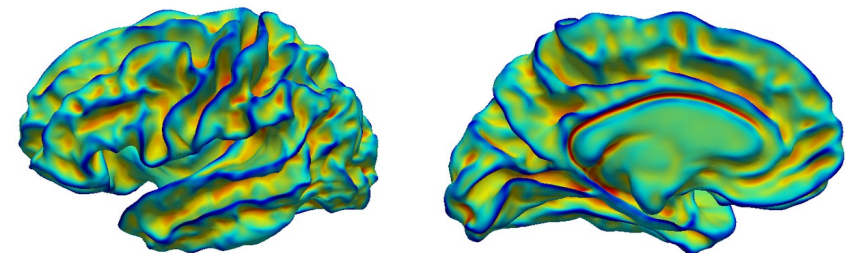
[Waehnert et al., Neuroimage 2014]  
[Tardif et al., Neuroimage 2015]  
[Rowley et al., Frontiers 2015]

# Multimodal alignment

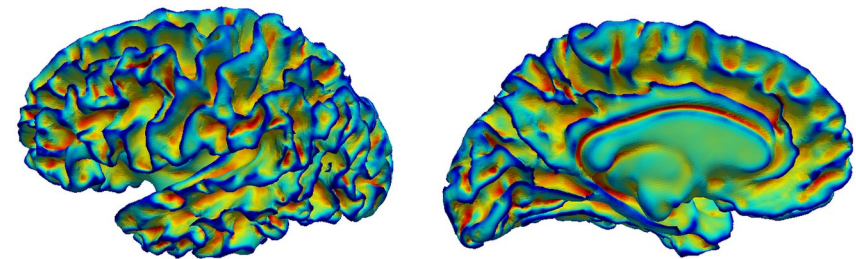
- Integrated linear and non-linear alignment (encapsulating ANTS)
- Scanner-based alignment (especially important for slab data)
- Transformation and deformation composition tools: register many times, transform once
- Multi-contrast cortical surface alignment



Slab data alignment



FreeSurfer



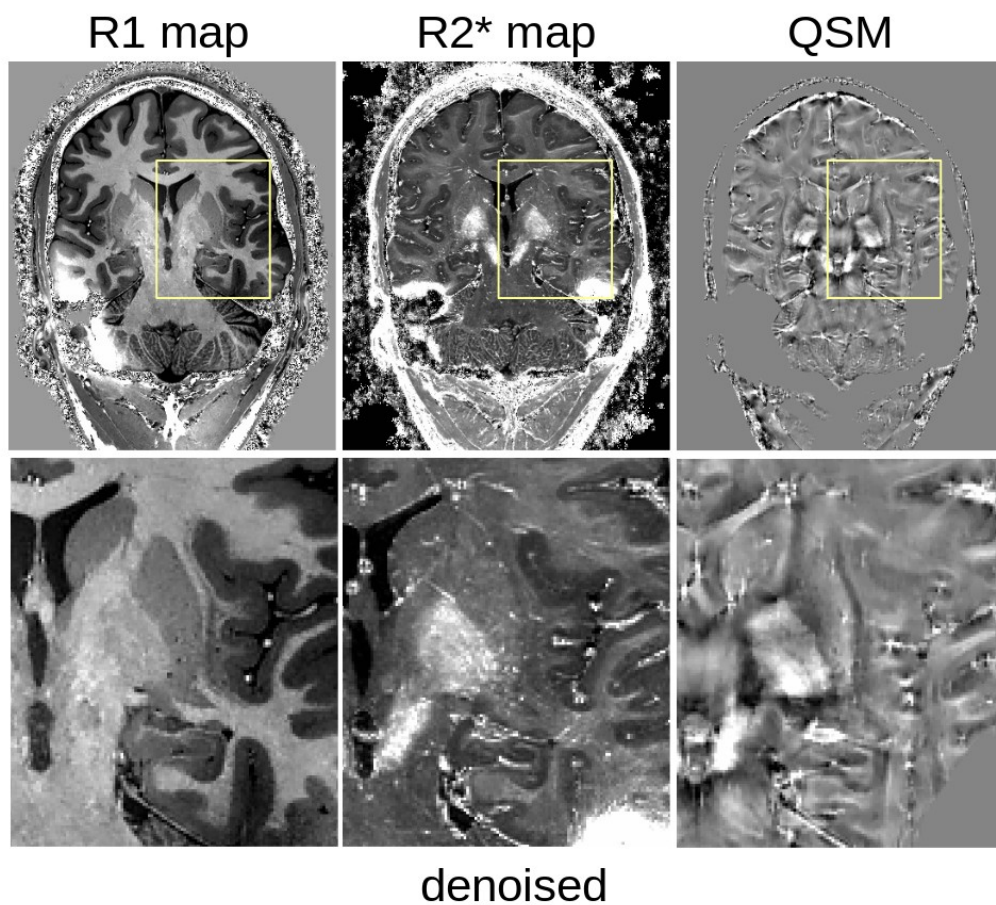
MMSR

Cortical surface registration

[Tardif et al., Neuroimage 2015]

# Quantitative MRI reconstruction & denoising

- MP2RAGE-based T1 mapping
- T2\*/R2\* model fitting
- PD ratio estimation
- Multi-contrast PCA denoising
- Fast phase unwarping

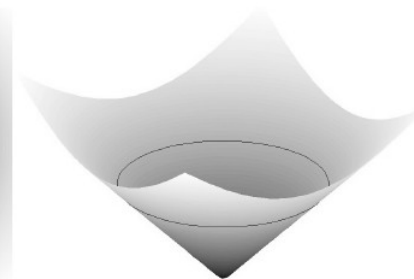
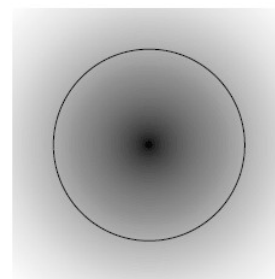
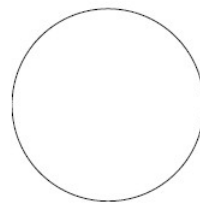


[Caan et al., 2018]  
[Bazin et al., 2019]

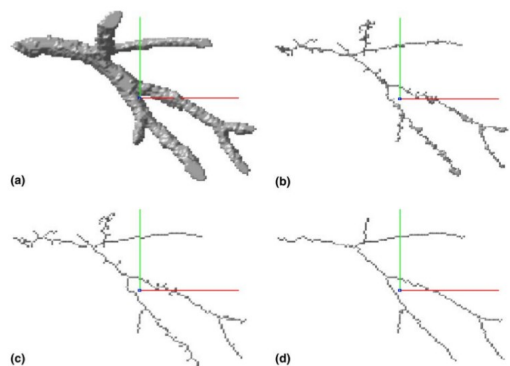


# Shape analysis & processing

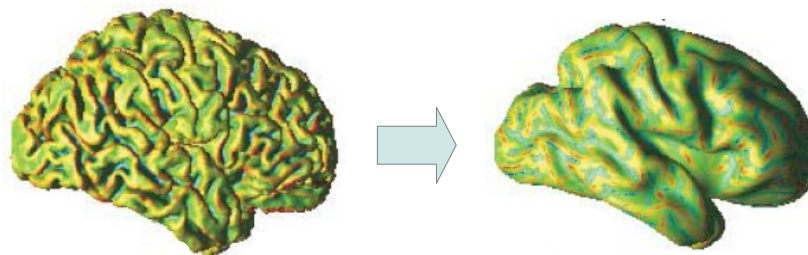
- Probability / levelset mapping
- Surface smoothing, inflation, meshing
- Skeletonization
- Topology correction
- Shape-based data propagation



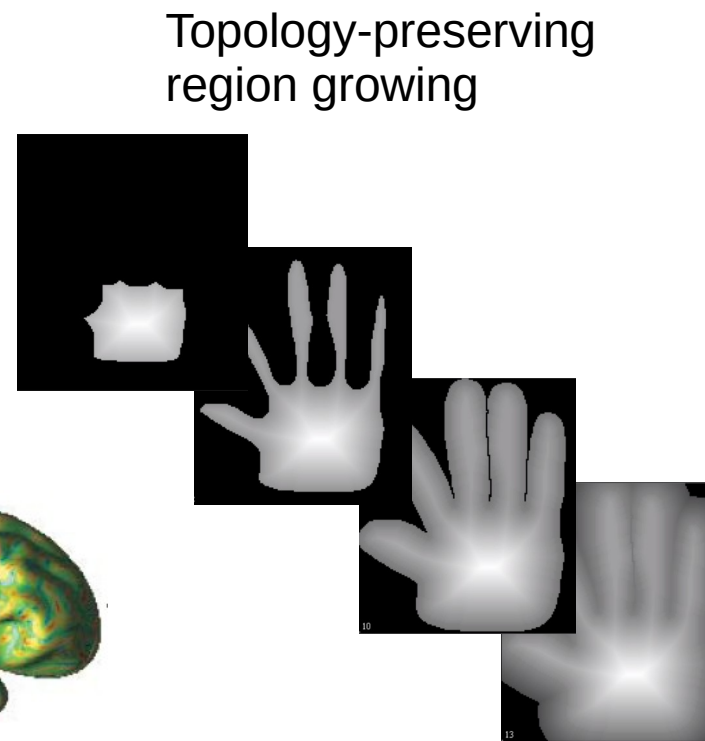
[Han et al., 2003]



[Bouix et al., 2005]



[Tosun et al., 2007]



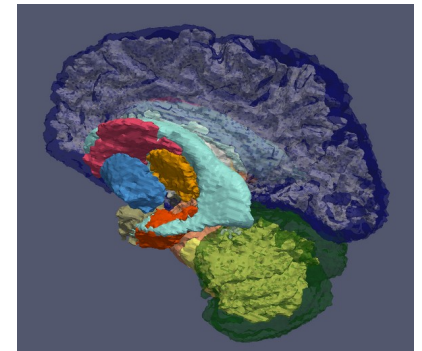
[Bazin et al., 2007]



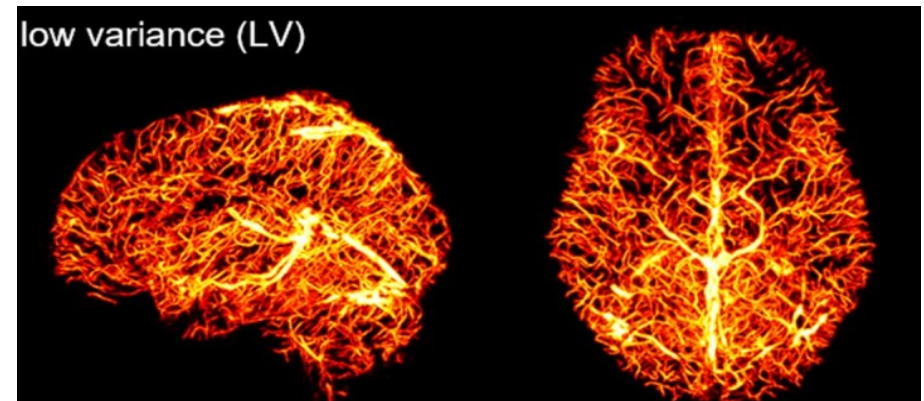
# Anatomical parcellation

- Whole brain / Cortex
- Cerebellum
- Subcortex
- Vasculature

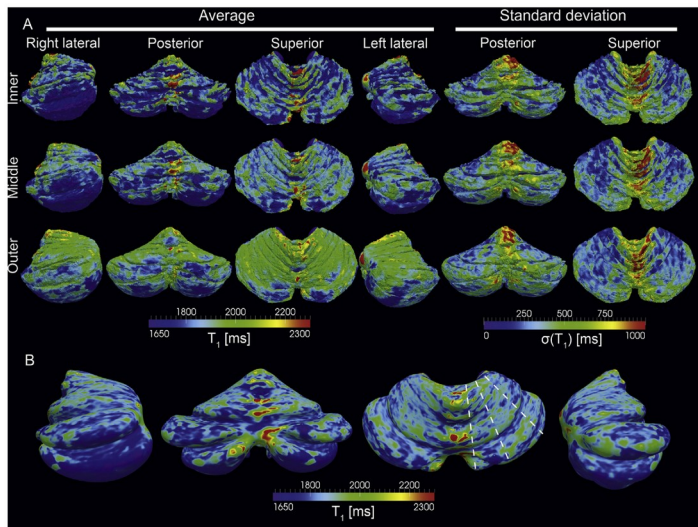
DWI segmentation



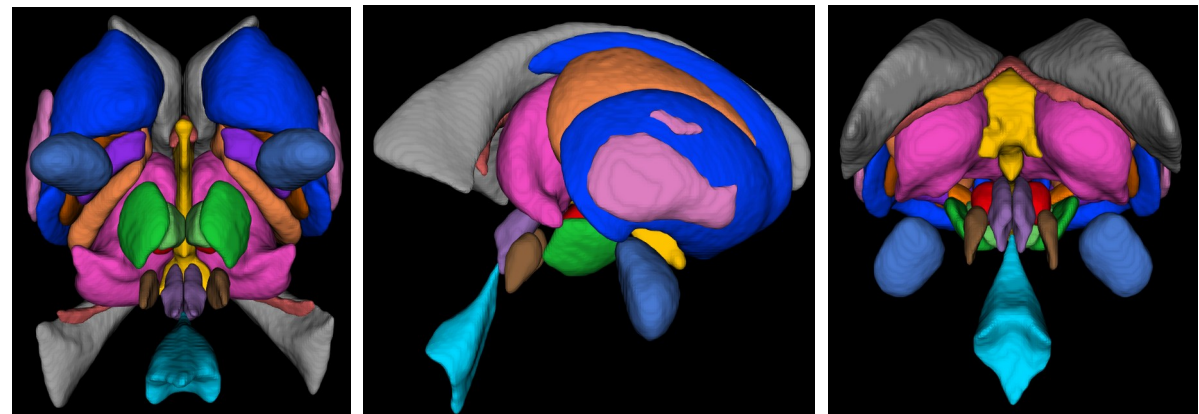
low variance (LV)



[Huck et al., 2019]



[Boillat et al., 2018]



[Bazin et al., 2020]

# The toolbox



Google  
Summer of Code



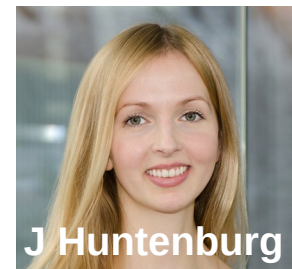
**[github.com/nighres/nighres](https://github.com/nighres/nighres)**

- Python wrappers for Java algorithms via JCC (optimized performance, code continuity)

# The toolbox



Google  
Summer of Code



## [github.com/nighres/nighres](https://github.com/nighres/nighres)

- Python wrappers for Java algorithms via JCC (optimized performance, code continuity)
- Standardized modular structure built on Nibabel images (pipeline behavior and popular image formats)
- Automated testing (Travis), documentation building (Sphinx)
- Docker containerization
- Advanced processing examples for 7T brain segmentation, cortical layering, normalization, ...
- Open source code, documentation, example scripts, and real data sets

[Huntenburg et al., 2018]

# The toolbox



Google  
Summer of Code



C Steele



J Huntenburg

## nighres.readthedocs.io

Nighres  
latest

Search docs

GETTING STARTED

- Installing Nighres
- Nighres usage examples

MODULES AND FUNCTIONS

- Brain
- Cortex
- Surface
- Laminar
- Data
- Input/Output

GOOD TO KNOW

- Data handling and the nimg
- Saving outputs
- Levelsets

DEVELOPER'S GUIDE

- Overview
- Setting up
- Wrapping an existing CBS Tools class
- Adding a new Python function
- Writing examples
- Adapting the docs
- Making a Pull Request

Docs » Welcome to Nighres!

[Edit on GitHub](#)

build passing docs passing

### Welcome to Nighres!

Nighres is a Python package for processing of high-resolution neuroimaging data. It developed out of CBS High-Res Brain Processing Tools and aims to make those tools easier to install, use and extend.

#### Warning

Nighres is currently still in beta stage

### Getting started

- Installing Nighres
- Nighres usage examples

### Modules and Functions

- Brain
  - mgdm\_segmentation
  - mp2rage\_skullstripping
  - extract\_brain\_region
- Cortex
  - cruise\_cortex\_extraction
- Surface
  - probability\_to\_levelset
- Laminar

Search or Jump to ... Pull requests Issues Marketplace Explore

nighres / nighres

Unwatch 8 Star 13 Fork 12

Code Issues 23 Pull requests 0 Projects 1 Wiki Insights Settings

Processing tools for high-resolution neuroimaging <http://nighres.readthedocs.io/en/latest/> Edit

Manage topics

331 commits 3 branches 9 releases 1 environment 6 contributors Apache-2.0

Your recently pushed branches:

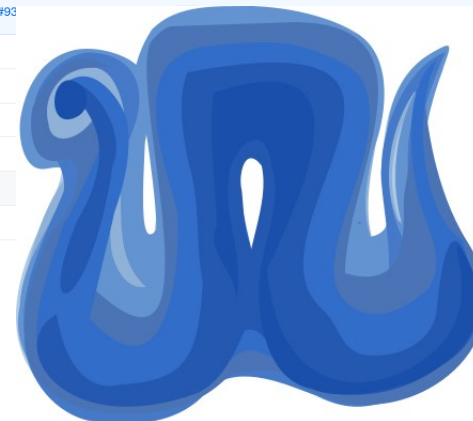
piloubazin:release-1.1.0b (39 minutes ago) Compare & pull request

Branch: master New pull request

Create new file Upload files Find file Clone or download

piloubazin Merge pull request #93

- cbstools
- doc
- docker
- examples
- nighres
- travis



# NIGHRES

Neuroimaging at  
high resolution



# The toolbox



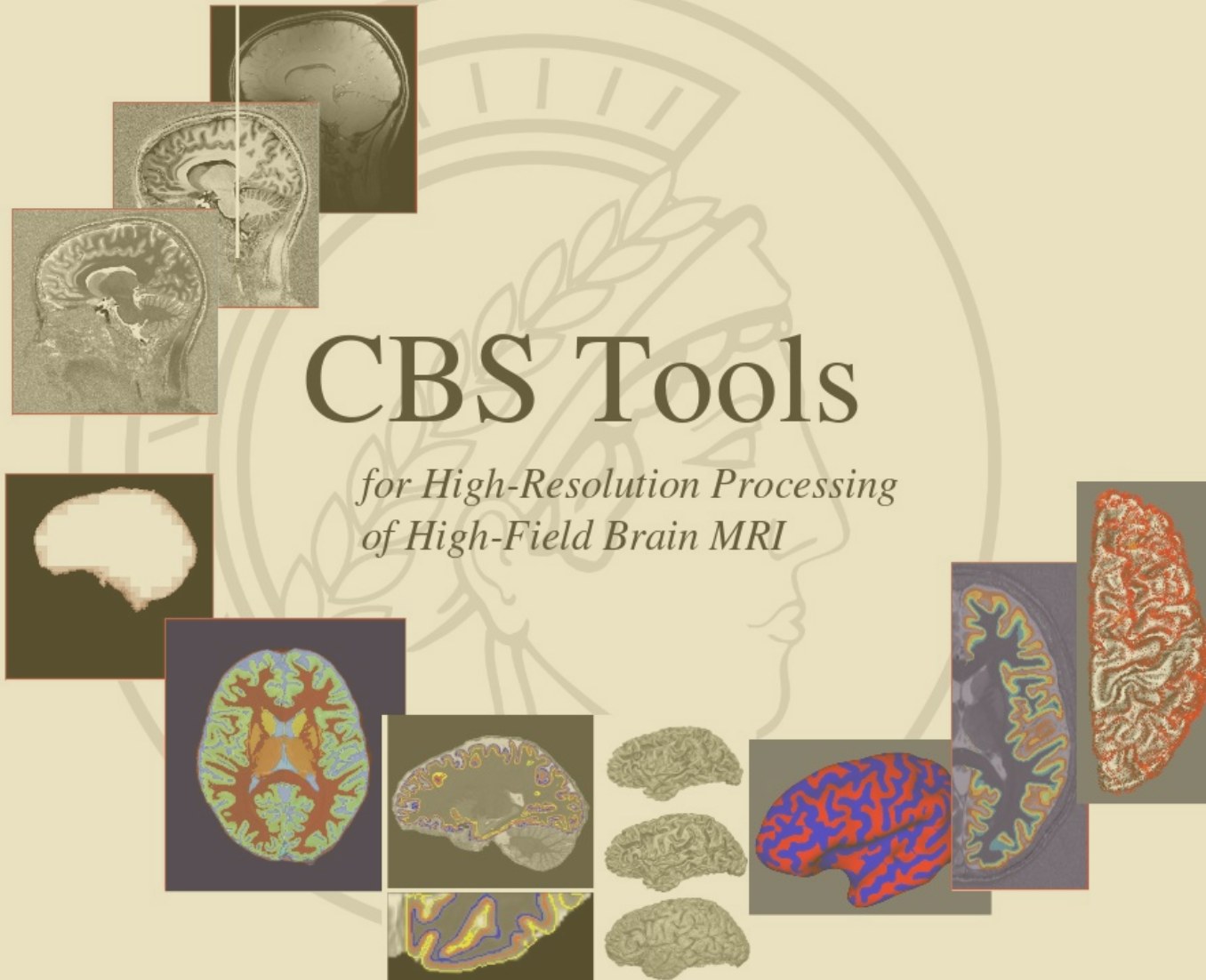
Google  
Summer of Code



**[github.com/nighres/nighres](https://github.com/nighres/nighres)**

- **Python wrappers for Java algorithms via JCC**  
(optimized performance, code continuity)
- Standardized modular structure built on Nibabel images  
(pipeline behavior and popular image formats)
- Automated testing (Travis), documentation building (Sphinx)
- Docker containerization
- Advanced processing examples for 7T brain segmentation,  
cortical layering, normalization, ...
- Open source code, documentation, example scripts,  
and real data sets

[Huntenburg et al., 2018]

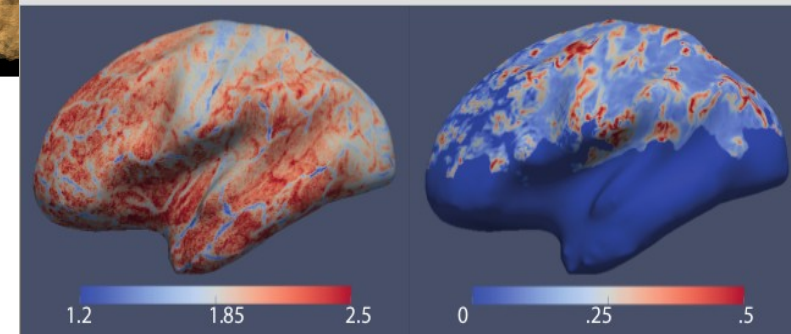
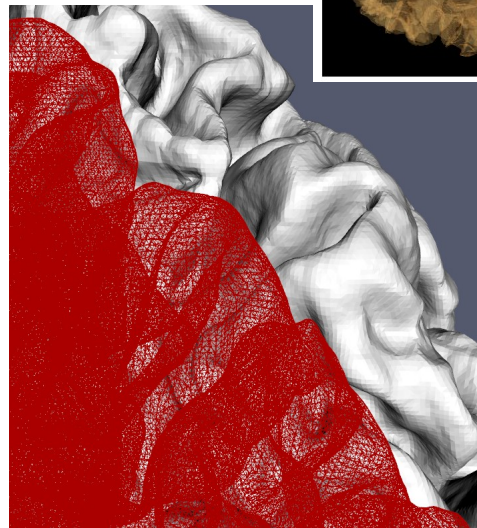
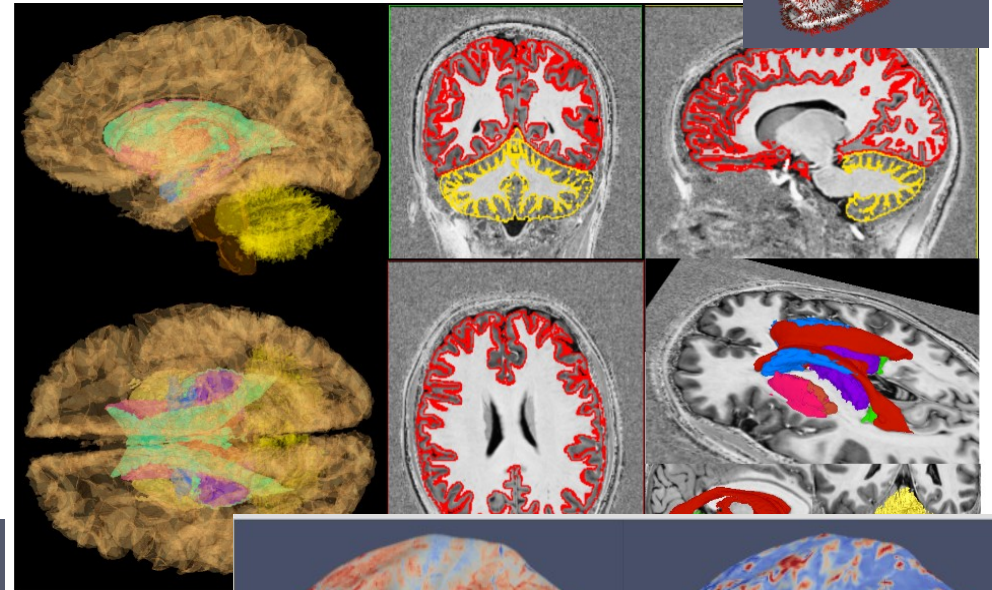
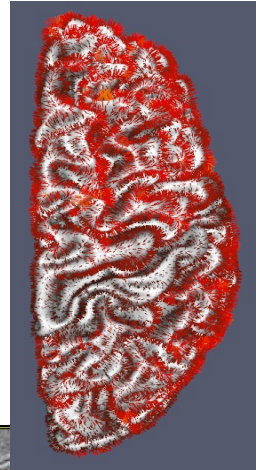
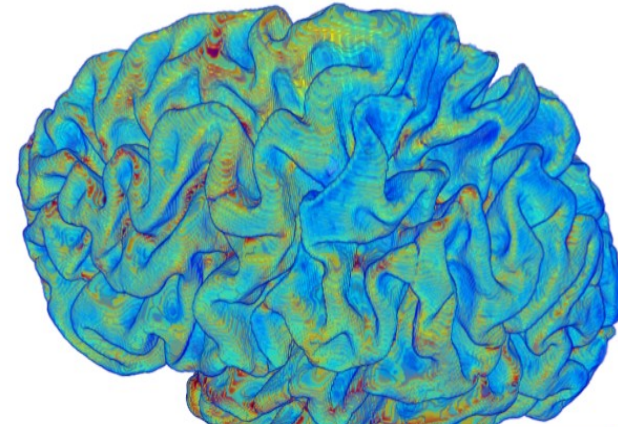


or: whatever happens to old toolboxes?

# What were the CBS Tools?

High resolution image processing algorithms **designed for 7T data**

- Handles MP2RAGE data, other **quantitative contrasts**
- **Segmentation**: cortical and sub-cortical structures
- Cortical **surface** extraction and inflation (cerebral and cerebellar)
- Processing in MNI space at **0.4 mm**
- Highly accurate cortical **layering, profiling** and **co-registration**



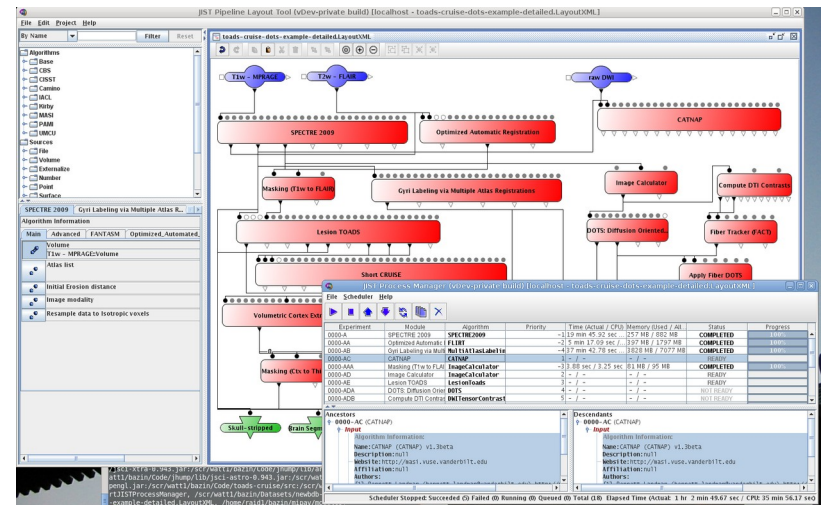
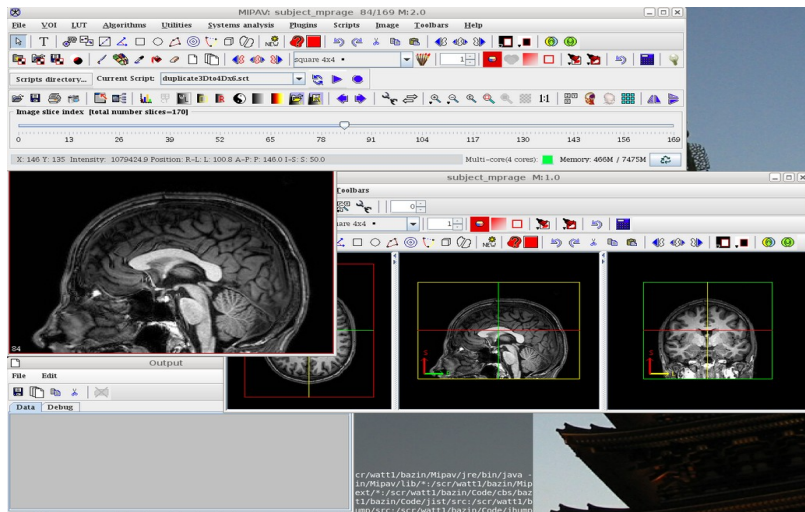
[Bazin et al., 2014]  
[Tardif et al., 2015]  
[Waehnert et al., 2016]



# How was the software organized?

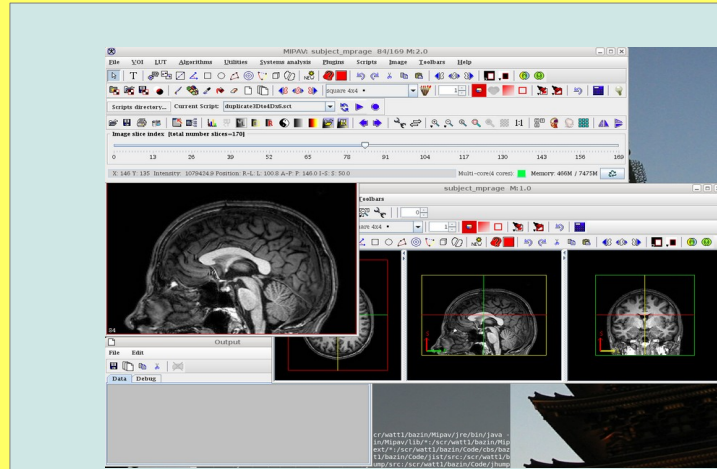
All algorithms are built as **plug-ins** for the **MIPAV** and **JIST** software:

- Support for **most medical image formats**
- User-friendly **graphical interface**
- Intuitive **pipeline system** for batch processing
- Integration with **many additional software tools**
- Ongoing **development and support** at participating institutions (NIH, MPI, JHU, Vanderbilt, ...)
- Open source and freely **available** in NITRC, NeuroDebian

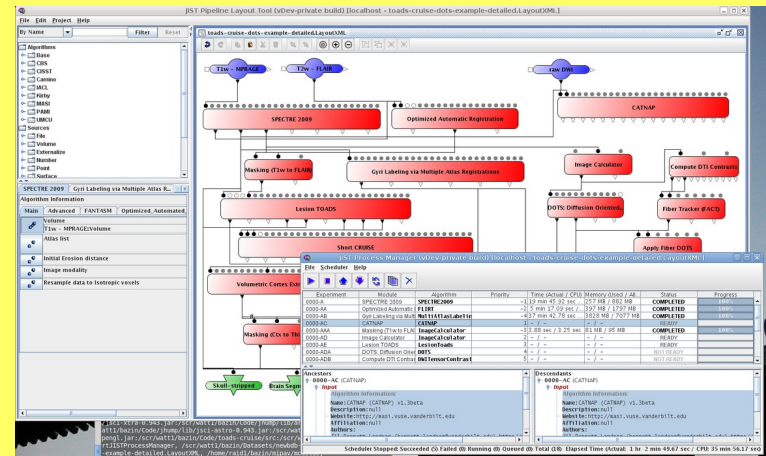


[McAuliffe et al., CBMS 2001]  
[Lucas et al., Neuroinf. 2010]

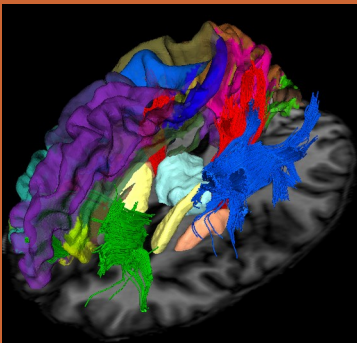
# How was the software organized?



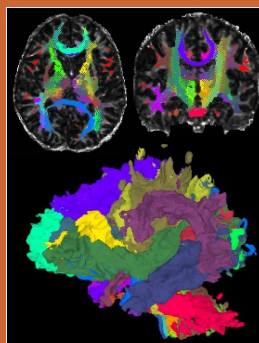
MIPAV: basic libraries, image visualization and editing



JIST: pipelining environment



CRUISE



DOTS

...



CBS TOOLS

# How did it work ?

## 1. Load a processing layout in the JIST layout manager

The screenshot displays the JIST Pipeline Layout Tool interface. The main window shows a detailed flowchart of a neuroimaging pipeline. The workflow starts with input volumes (T1w - MPRAGE and T2w - FLAIR) and raw DWI data. Key processing steps include SPECTRE 2009, Optimized Automatic Registration, CATNAP, Masking (T1w to FLAIR), Gyri Labeling via Multiple Atlas Registrations, Image Calculator, Compute DTI Contrasts, Lesion TOADS, DOTS: Diffusion Oriented..., Fiber Tracker (FACT), Short CRUISE, Apply Fiber DOTS, Volumetric Cortex Extraction, and Masking (Ctx to Th). The final outputs are Skull-stripped and Brain Segmentation.

Overlaid on the pipeline is the JIST Process Manager window, which provides a detailed view of the execution progress. The table below summarizes the tasks shown in the manager:

Experiment	Module	Algorithm	Priority	Time (Actual / CPU)	Memory (Used / All...)	Status	Progress
0000-A	SPECTRE 2009	SPECTRE2009	-1	19 min 45.92 sec ...	257 MB / 882 MB	COMPLETED	100%
0000-AA	Optimized Automatic	FLIRT	-2	5 min 17.09 sec / ...	397 MB / 1797 MB	COMPLETED	100%
0000-AB	Gyri Labeling via Multi	MultiAtlasLabelin	-4	37 min 42.78 sec ...	3828 MB / 7077 MB	COMPLETED	100%
0000-AC	CATNAP	CATNAP	1	- / -	- / -	READY	-
0000-AAA	Masking (T1w to FLAI	ImageCalculator	-3	3.88 sec / 3.25 sec	81 MB / 95 MB	COMPLETED	100%
0000-AD	Image Calculator	ImageCalculator	2	- / -	- / -	READY	-
0000-AE	Lesion TOADS	LesionToads	3	- / -	- / -	READY	-
0000-ADA	DOTS: Diffusion Orie	DOTS	4	- / -	- / -	NOT READY	-
0000-ADB	Compute DTI Contra	DwtensorContrast	5	- / -	- / -	NOT READY	-

At the bottom of the JIST Process Manager window, the scheduler status is reported: Scheduler Stopped: Succeeded (6) Failed (0) Running (0) Queued (0) Total (18) Elapsed Time (Actual): 1 hr 2 min 49.67 sec / CPU: 35 min 56.17 sec.



# How did it work ?

## 2. Input your data

The screenshot displays the JIST Pipeline Layout Tool interface. The main window shows a workflow diagram with nodes for data input, processing, and output. The process manager window is open, showing a table of tasks and their execution status.

Experiment	Module	Algorithm	Priority	Time (Actual / CPU)	Memory (Used / All..)	Status	Progress
0000-A	SPECTRE 2009	SPECTRE2009		-1 19 min 45.92 sec ...	257 MB / 882 MB	COMPLETED	100%
0000-AA	Optimized Automatic	FLIRT		-2 5 min 17.09 sec / ...	397 MB / 1797 MB	COMPLETED	100%
0000-AB	Gyri Labeling via Multi	MultiAtlasLabelin		-4 37 min 42.78 sec ...	3828 MB / 7077 MB	COMPLETED	100%
0000-AC	CATNAP	CATNAP	1	- / -	- / -	READY	
0000-AAA	Masking (T1w to FLAI	ImageCalculator		-3 3.88 sec / 3.25 sec	81 MB / 95 MB	COMPLETED	100%
0000-AD	Image Calculator	ImageCalculator	2	- / -	- / -	READY	
0000-AE	Lesion TOADS	LesionToads	3	- / -	- / -	READY	
0000-ADA	DOTS: Diffusion Orient	DOTS	4	- / -	- / -	NOT READY	
0000-ADB	Compute DTI Contra	DwtensorContrast	5	- / -	- / -	NOT READY	

At the bottom of the process manager window, the scheduler status is shown: Scheduler Stopped: Succeeded (6) Failed (0) Running (0) Queued (0) Total (18) Elapsed Time (Actual): 1 hr 2 min 49.67 sec / CPU: 35 min 56.17 sec

# How did it work ?

## 3. Open the JIST Process Manager

The screenshot displays the JIST Pipeline Layout Tool (vDev-private build) and the JIST Process Manager (vDev-private build). The top window shows a pipeline diagram with steps like SPECTRE 2009, CATNAP, and Fiber Tracker. The bottom window shows a table of process execution details.

Experiment	Module	Algorithm	Priority	Time (Actual / CPU)	Memory (Used / All..)	Status	Progress
0000-A	SPECTRE 2009	SPECTRE2009		-1 19 min 45.92 sec ...	257 MB / 882 MB	COMPLETED	100%
0000-AA	Optimized Automatic	FLIRT		-2 5 min 17.99 sec / ...	397 MB / 1797 MB	COMPLETED	100%
0000-AB	Gyri Labeling via Multi	MultiAtlasLabelin		-4 37 min 42.78 sec ...	3828 MB / 7077 MB	COMPLETED	100%
0000-AC	CATNAP	CATNAP	1	- / -	- / -	READY	
0000-AAA	Masking (T1w to FLAI	ImageCalculator		-3 3.88 sec / 3.25 sec	81 MB / 95 MB	COMPLETED	100%
0000-AD	Image Calculator	ImageCalculator	2	- / -	- / -	READY	
0000-AE	Lesion TOADS	LesionToads	3	- / -	- / -	READY	
0000-ADA	DOTS: Diffusion Orie	DOTS	4	- / -	- / -	NOT READY	
0000-ADB	Compute DTI Contra	DwtensorContrast	5	- / -	- / -	NOT READY	

Process Manager Summary: Scheduler Stopped: Succeeded (6) Failed (0) Running (0) Queued (0) Total (18) Elapsed Time (Actual: 1 hr 2 min 49.67 sec, CPU: 35 min 56.17 sec)

# How did it work?

The screenshot displays the JIST Pipeline Layout Tool interface. The main window shows a workflow diagram with nodes for 'SPECTRE 2009', 'Optimized Automatic Registration', 'CATNAP', 'Masking (T1w to FLAIR)', 'Gyri Labeling via Multiple Atlas Registrations', 'Image Calculator', 'Compute DTI Contrasts', 'Lesion TOADS', 'DOTS: Diffusion Oriented...', 'Fiber Tracker (FACT)', 'Short CRUISE', 'Apply Fiber DOTS', and 'Volumetric Cortex Extr'. A blue circle highlights the 'Volumetric Cortex Extr' node, with an arrow pointing to the 'JIST Process Manager' window below it.

The 'JIST Process Manager' window shows a table of process execution details:

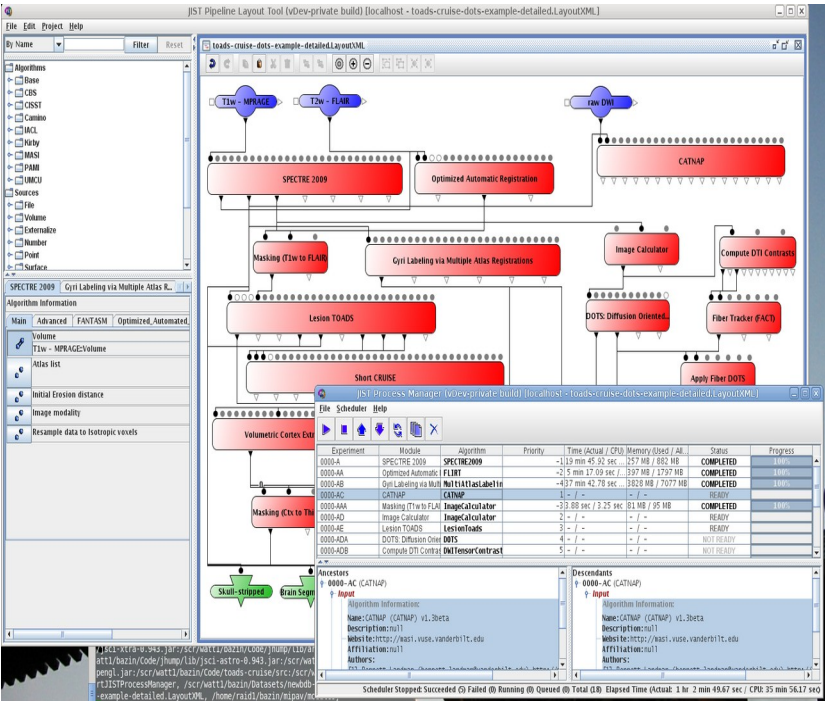
Experiment	Module	Algorithm	Priority	Time (Actual / CPU)	Memory (Used / All..)	Status	Progress
0000-AA	SPECTRE 2009	SPECTRE2009		-1 19 min 45.92 sec ...	257 MB / 882 MB	COMPLETED	100%
0000-AB	Optimized Automatic	FLIRT		-2 5 min 17.99 sec / ...	397 MB / 1797 MB	COMPLETED	100%
0000-AC	Gyri Labeling via Multi	MultiAtlasLabelin		-4 37 min 42.78 sec ...	3828 MB / 7077 MB	COMPLETED	100%
0000-AD	CATNAP	CATNAP	1	- / -	- / -	READY	
0000-AA	Masking (T1w to FLAI	ImageCalculator		-3 3.88 sec / 3.25 sec	81 MB / 95 MB	COMPLETED	100%
0000-AD	Image Calculator	ImageCalculator	2	- / -	- / -	READY	
0000-AE	Lesion TOADS	LesionToads	3	- / -	- / -	READY	
0000-ADA	DOTS: Diffusion Orie	DOTS	4	- / -	- / -	NOT READY	
0000-ADB	Compute DTI Contra	DwtTensorContrast	5	- / -	- / -	NOT READY	

At the bottom of the process manager window, a status bar indicates: 'Scheduler Stopped: Succeeded (6) Failed (0) Running (0) Queued (0) Total (18) Elapsed Time (Actual: 1 hr 2 min 49.67 sec / CPU: 35 min 56.17 sec)'

4. Press play

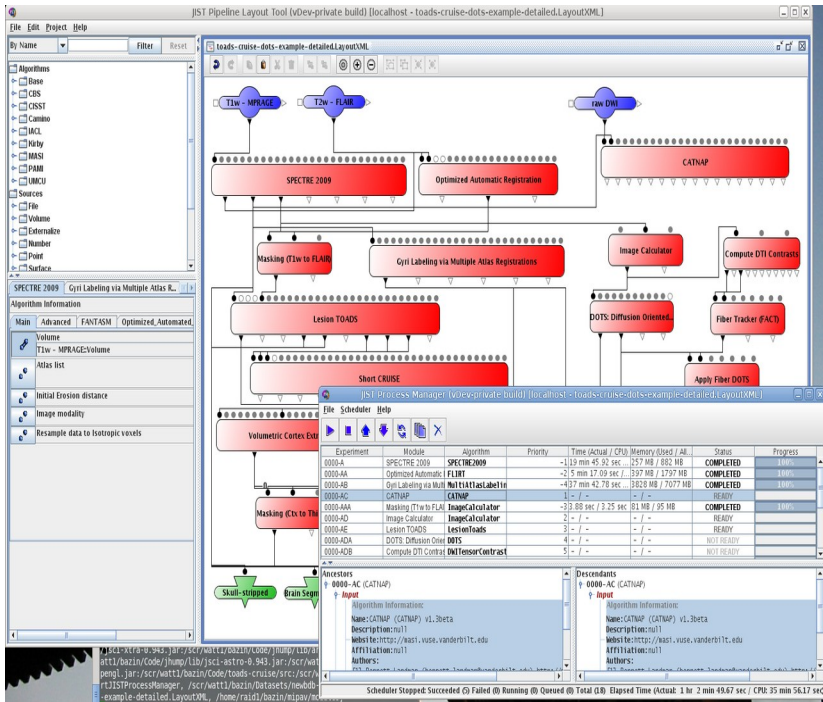


# Could the software adapt?

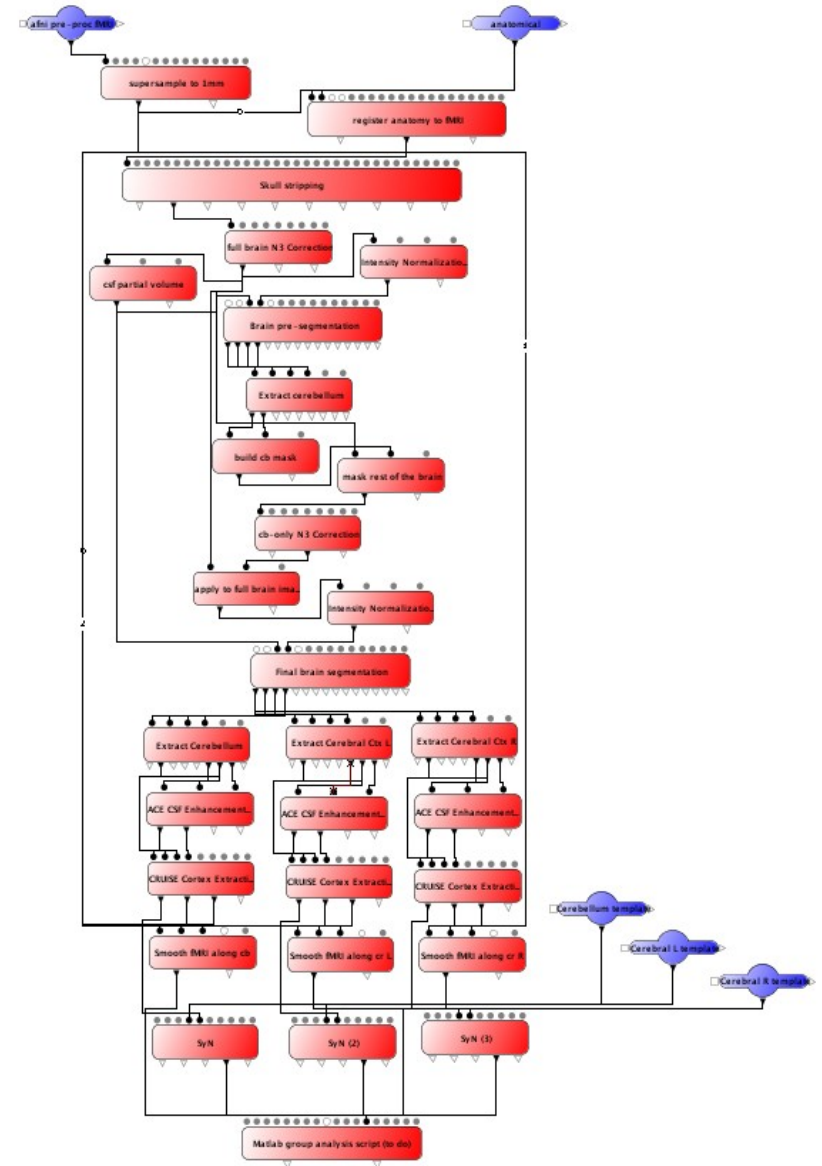
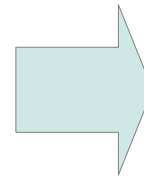


Standard pipelines

# Could the software adapt?



Standard pipelines



Customized pipelines

A perfect solution...?



# A perfect solution...?



Installation *dependencies*: MIPAV (easy), JIST (not easy)



External tool encapsulation possible, but complicated



JAVA-based (efficient, cross-platform, but not *popular*)

# A perfect solution...?



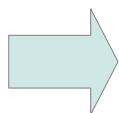
Installation dependencies: MIPAV (easy), JIST (not easy)



External tool encapsulation possible, but complicated



JAVA-based (efficient, cross-platform, but not *popular*)



Alternative: Neuroimaging in Python initiatives



Python-based (*hugely* popular)



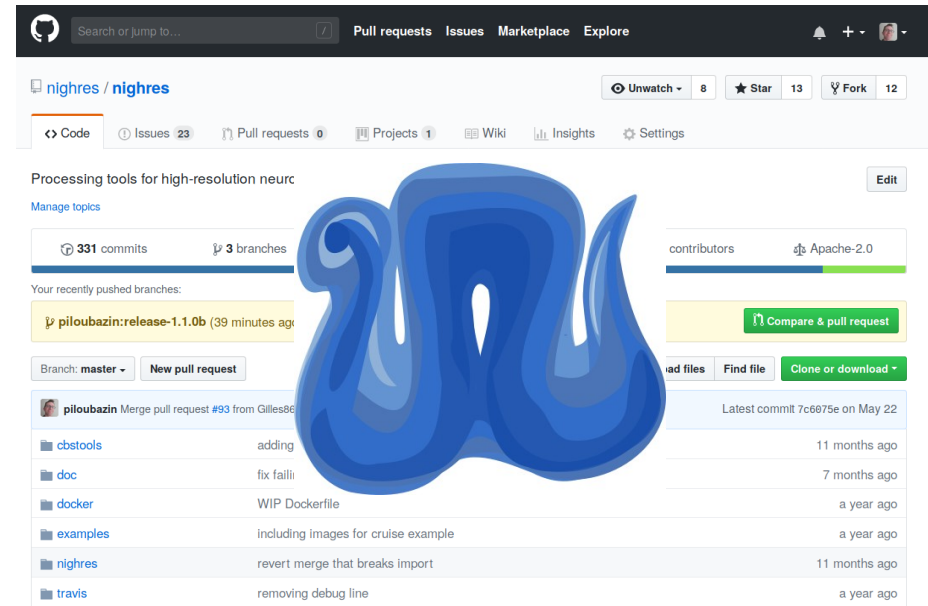
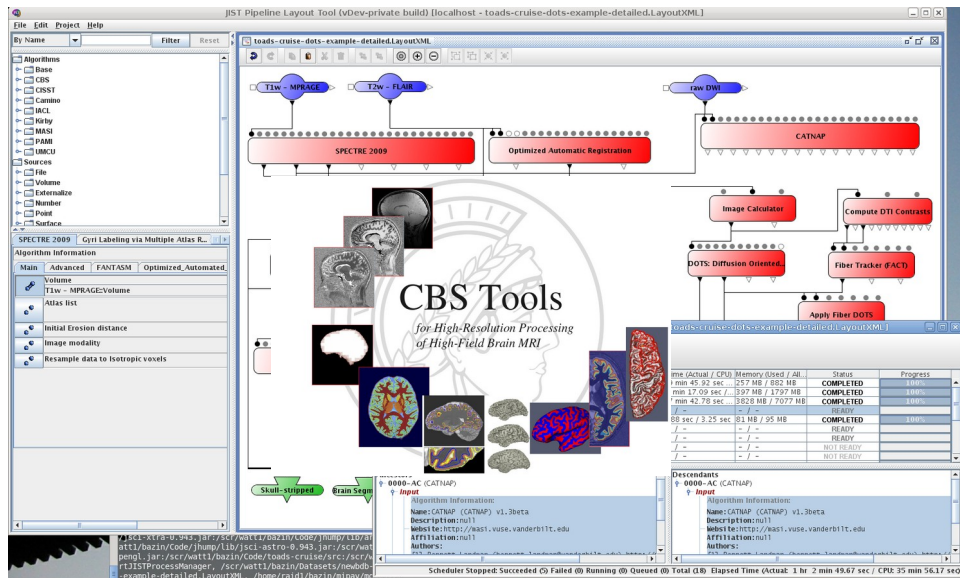
All the *cool* neuroimaging software is (wrapped) in Python



Massive online *user community*



# From CBS Tools to Nighres



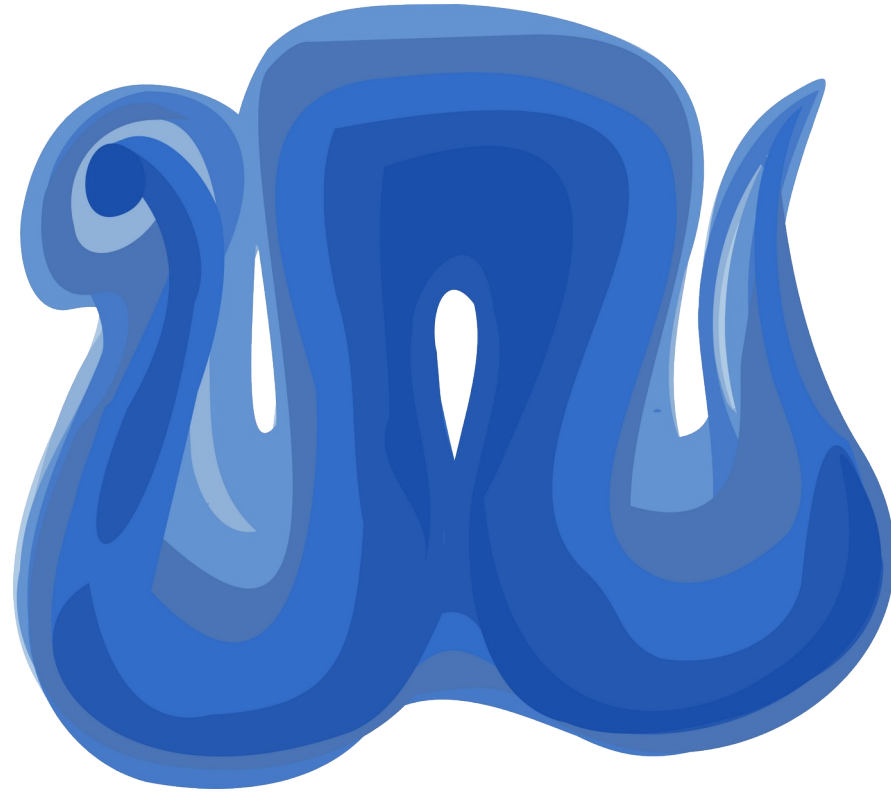
Open scientific tools: open source code is not enough...

Increasing *ease of use* may include:

- Getting rid of user interfaces (back to old-fashioned scripting)
- Providing interfaces to competing tools
- Complying with today's software *fashion* (python, docker, github...) to benefit from today's software *infrastructure* and online community



# Installation procedure(s)



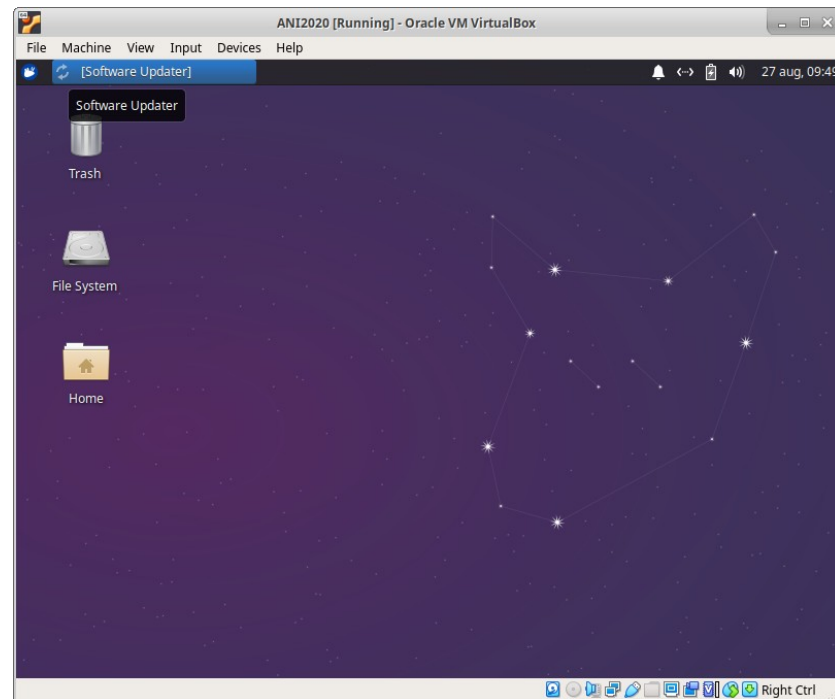
# Installation instructions

Virtual box installation for trial  
(docker also an option)

Preferred environment:  
UNIX system (Linux / MacOS)

Main dependencies:

- Python3 (numpy, pandas, nibabel...)
- Java 8+ JDK (Java developer kit)
- JCC (Java to C compiler for python integration)
- ANTs for registration routines



# Under the hood

Nighres modules:

python routines

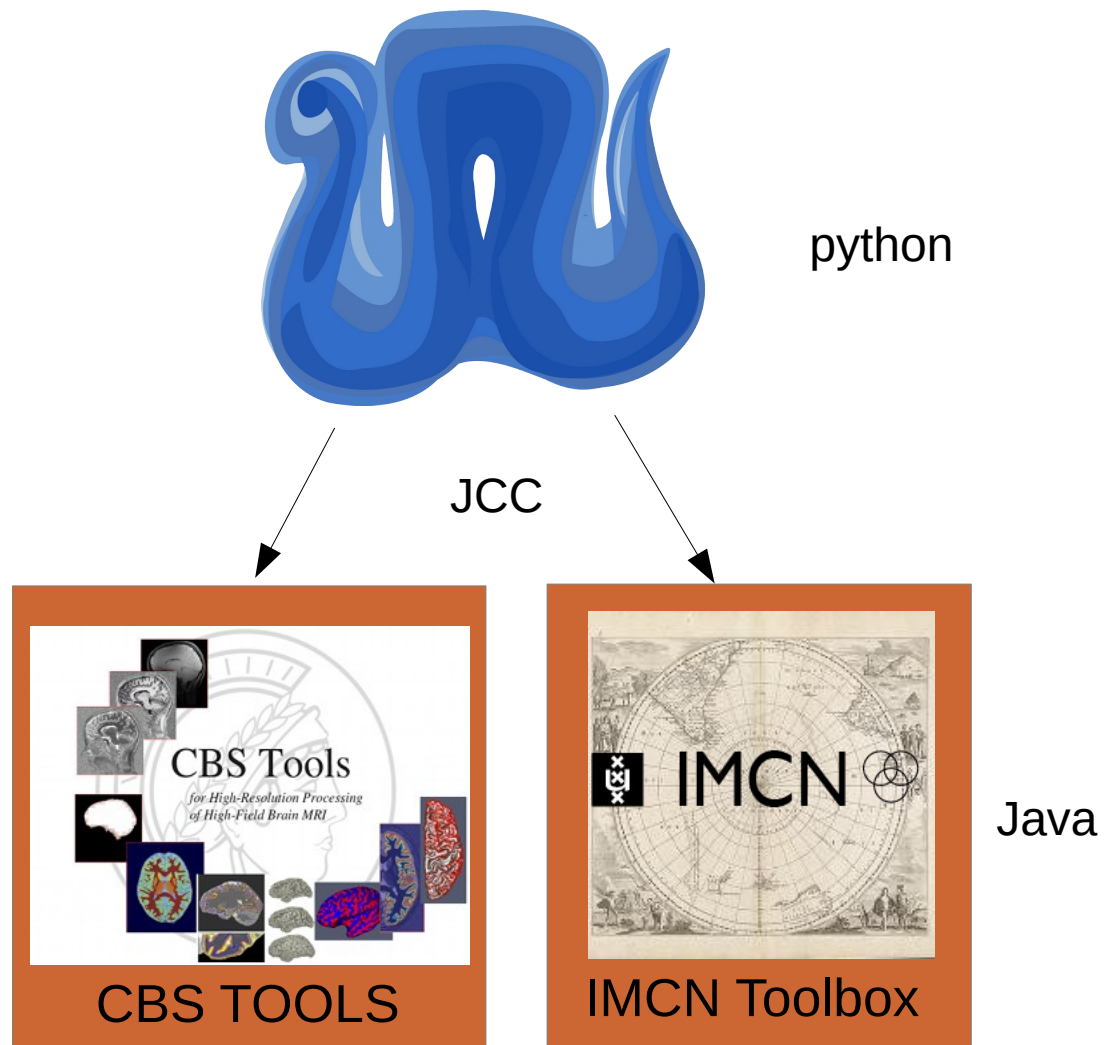
+ Java wrappers

+ ANTs wrappers



Python to Java calls

ANTs calls (exec)





# Common issues

python2 – python3 rivalry

Multiple versions of python3.x installed

Anaconda / Miniconda python

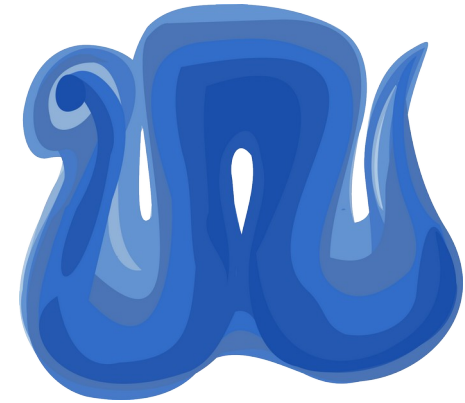
Java *JDK* not installed

JCC not finding the JDK

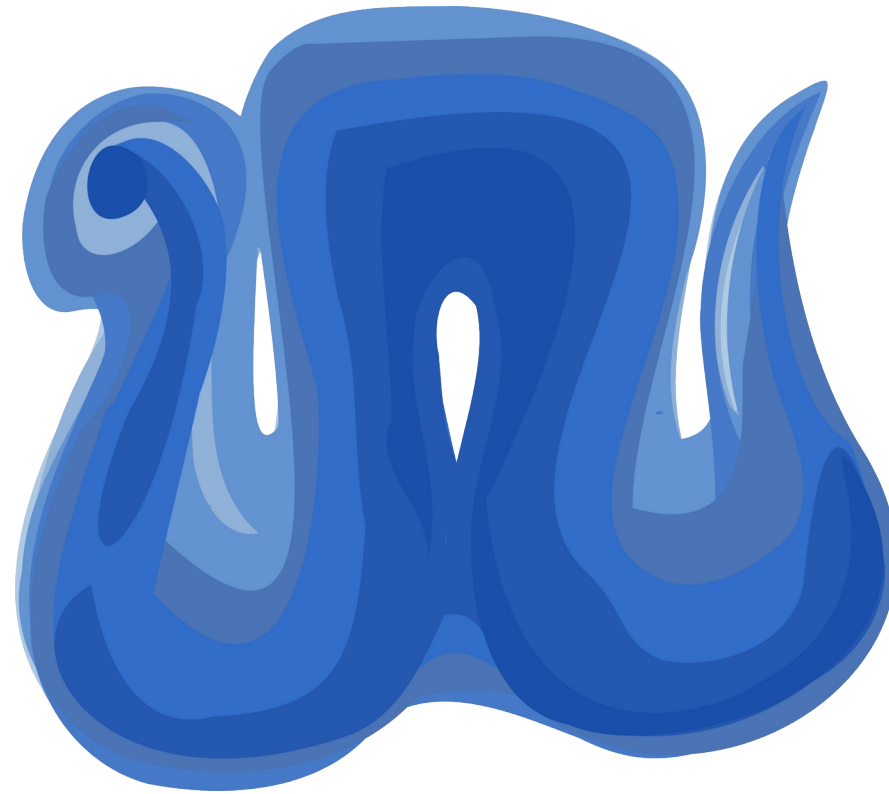
Compatibility issues of helper software (pip, docker, ...)

ANTs binaries not on the Unix PATH variable

Memory limitations on high resolution data

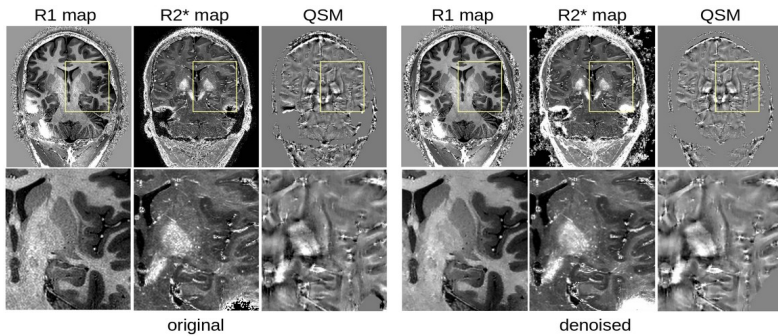


# Nighres modules step-by-step

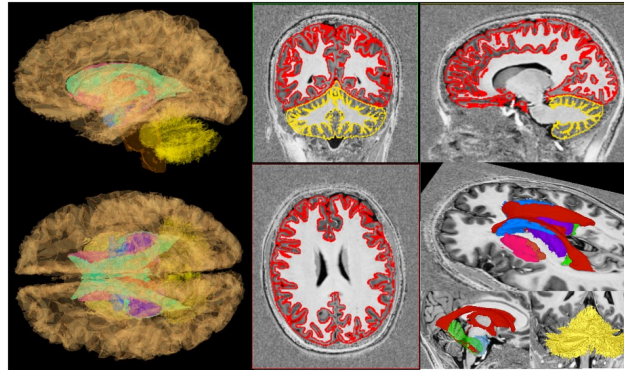


# Modules, modules, modules...

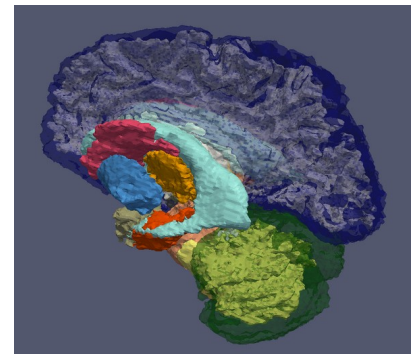
Quantitative MRI reconstruction and denoising



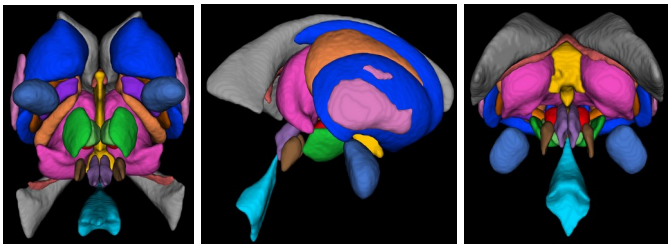
MP2RAGE Whole brain segmentation



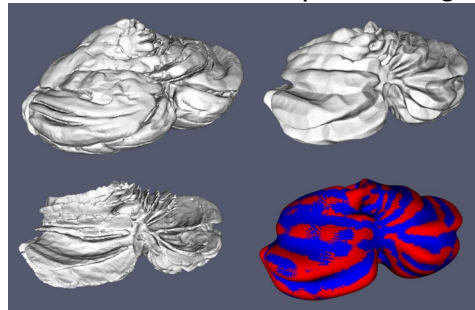
DWI segmentation



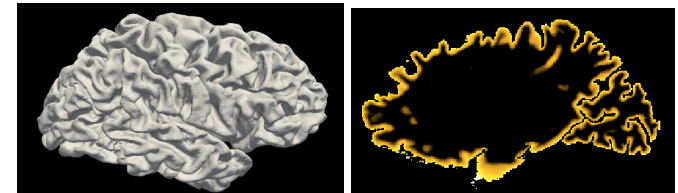
Subcortical Parcellation



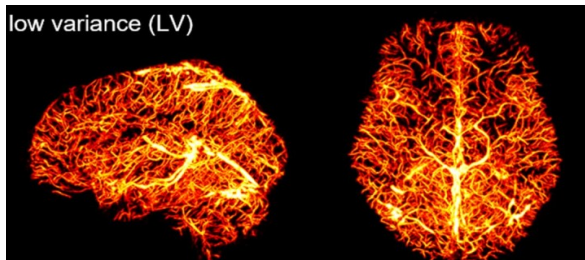
Cerebellar surface processing



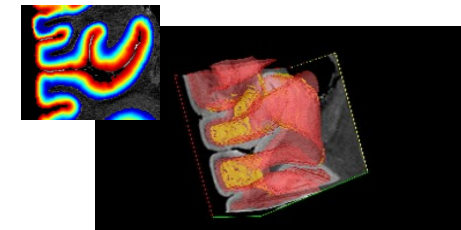
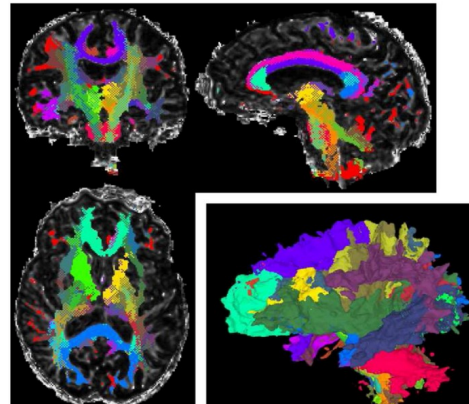
Cortical & laminar analysis



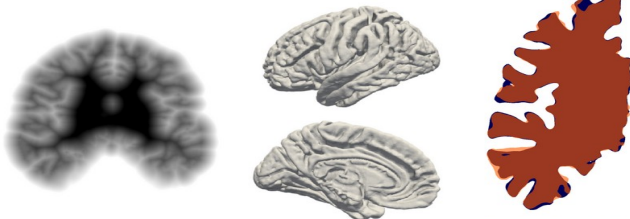
Vascular reconstruction



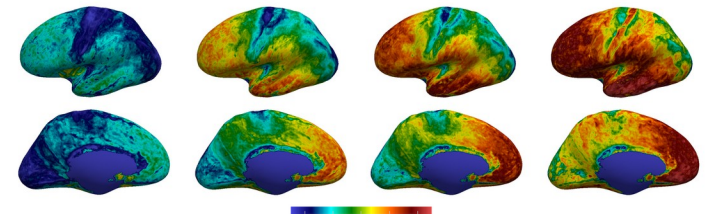
White matter tract labelling



Surface-based registration

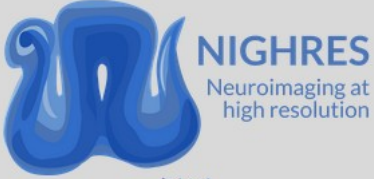


Volume-preserving lamination





# General module anatomy



latest

GETTING STARTED

- Installing Nighres
- Nighres usage examples

MODULES AND FUNCTIONS

- Brain
- Cortex
- Data
- Filtering
- Intensity
- Input/Output
- Laminar
- Microscopy
- Registration
- Segmentation
- Shape
- Statistics

Surface

Read the Docs

Docs » Surface » probability\_to\_levelset

[Edit on GitHub](#)

## probability\_to\_levelset

```
nighres.surface.probability_to_levelset(probability_image, mask_image=None, save_data=False, overwrite=False, output_dir=None, file_name=None) \[source\]
```

Levelset from probability map

Creates a levelset surface representations from a probabilistic map or a mask. The levelset indicates each voxel's distance to the closest boundary. It takes negative values inside and positive values outside of the object.

- Parameters:
- **probability\_image** (*niimg*) - Probability image to be turned into levelset. Values should be in [0, 1], either a binary mask or defining the boundary at 0.5.
  - **mask\_image** (*niimg, optional*) - Mask image defining the region in which to compute the levelset. Values equal to zero are set to maximum distance.
  - **save\_data** (*bool, optional*) - Save output data to file (default is False)
  - **overwrite** (*bool, optional*) - Overwrite existing results (default is False)
  - **output\_dir** (*str, optional*) - Path to desired output directory, will be created if it doesn't exist
  - **file\_name** (*str, optional*) - Desired base name for output files with file extension (suffixes will be added)

- Returns:
- Dictionary collecting outputs under the following keys (suffix of output files in brackets)
- **result** (*niimg*): Levelset representation of surface (*\_p2l-surf*)

Return type: `dict`

Main inputs  
(generally a niimg)

Options & parameters  
(preset)

Data saving options  
(standard)

Output dictionary  
(generally niimgs)

When saving to disk:

- modules pass file names rather than niimg objects
- data is not recomputed if file name exists

# Modules of particular interest

Nihres.brain.mgdm\_segmentation

Nihres.parcellation.massp

Nihres.brain.extract\_brain\_region

Nihres.cortex.cruise\_cortex\_extraction

Nihres.laminar.volumetric\_layering

Nihres.laminar.laminar\_iterative\_smoothing

Nihres.laminar.profile\_sampling

Nihres.laminar.profile\_meshing

Nihres.registration.embedded\_antsreg

Nihres.registration.apply\_coordinate\_mappings

Nihres.surface.surface\_inflation

Nihres.brain.intensity\_based\_skullstripping

Nihres.intensity.background\_estimation

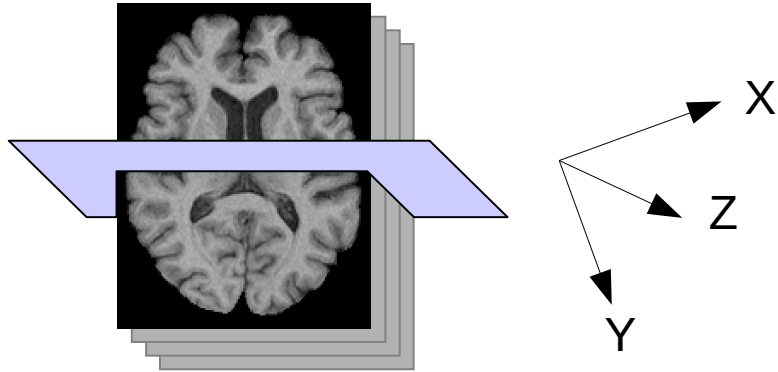
Nihres.intensity.lcpca\_denoising

Nihres.intensity.mp2rage\_t1\_mapping

Nihres.intensity.intensity\_propagation

# Important concepts: coordinate mappings

Image coordinate systems



Coordinate transforms between scanner and voxel spaces:

Stored in **image header**

NIFTI format: S-form, Q-form

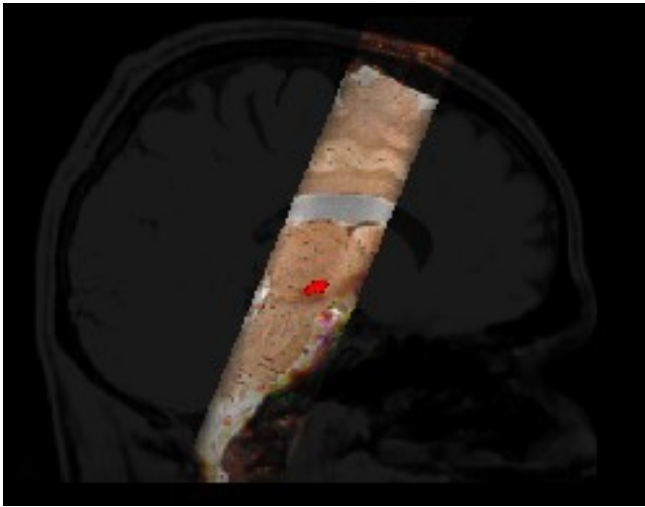
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} e_{11} & e_{12} & e_{13} & e_{14} \\ e_{21} & e_{22} & e_{23} & e_{24} \\ e_{31} & e_{32} & e_{33} & e_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Scanner (world) coordinates

Voxels



# Aligning images



Images from the same scanning session have common *scanner coordinates*

➡ Pre-computed rigid transform (S/Q-form)

Subject may still move slightly between scans

➡ Rigid transformation

Distortions?

1. Correct from additional data when possible (reversed encoding in DWI, fieldmaps in fMRI)
2. Non-linear registration (with ANTS)

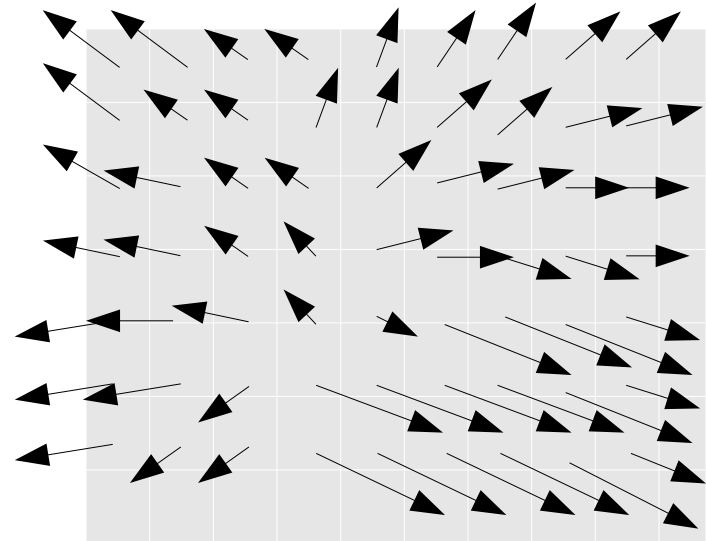
➡ Non-linear deformation

# The problem: combining transformations

Linear transformations: 4x4 matrix

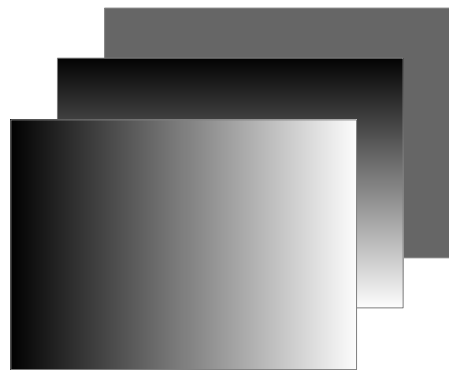
$R_{xx}$	$R_{xy}$	$R_{xz}$	$T_x$
$R_{yx}$	$R_{yy}$	$R_{yz}$	$T_y$
$R_{zx}$	$R_{zy}$	$R_{zz}$	$T_z$
0	0	0	1

Non-linear deformations: vector fields



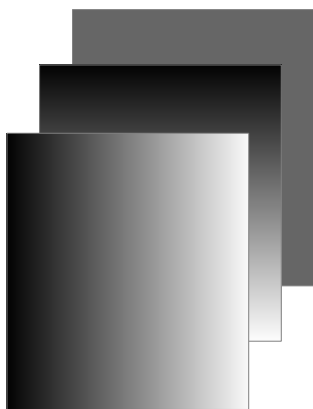
In which space?

# Introducing: coordinate mappings



Coordinate mapping:  
values  $(X, Y, Z)$  **subject**  
in **target** space

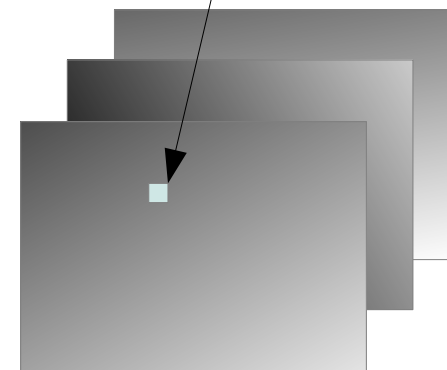
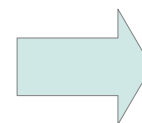
Combining deformations:



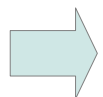
Subject to target 1



Target 1 to target 2



Subject to target 2

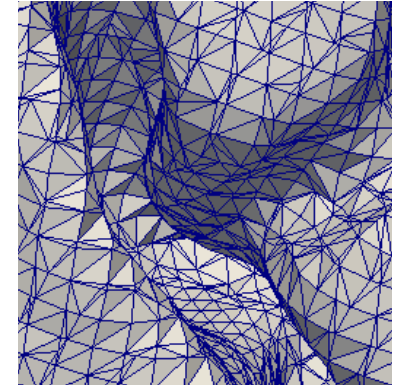


- Mappings have same dimensions as target
- Transformations can be checked by looking at coordinate values
- Combining transformations from different spaces easier

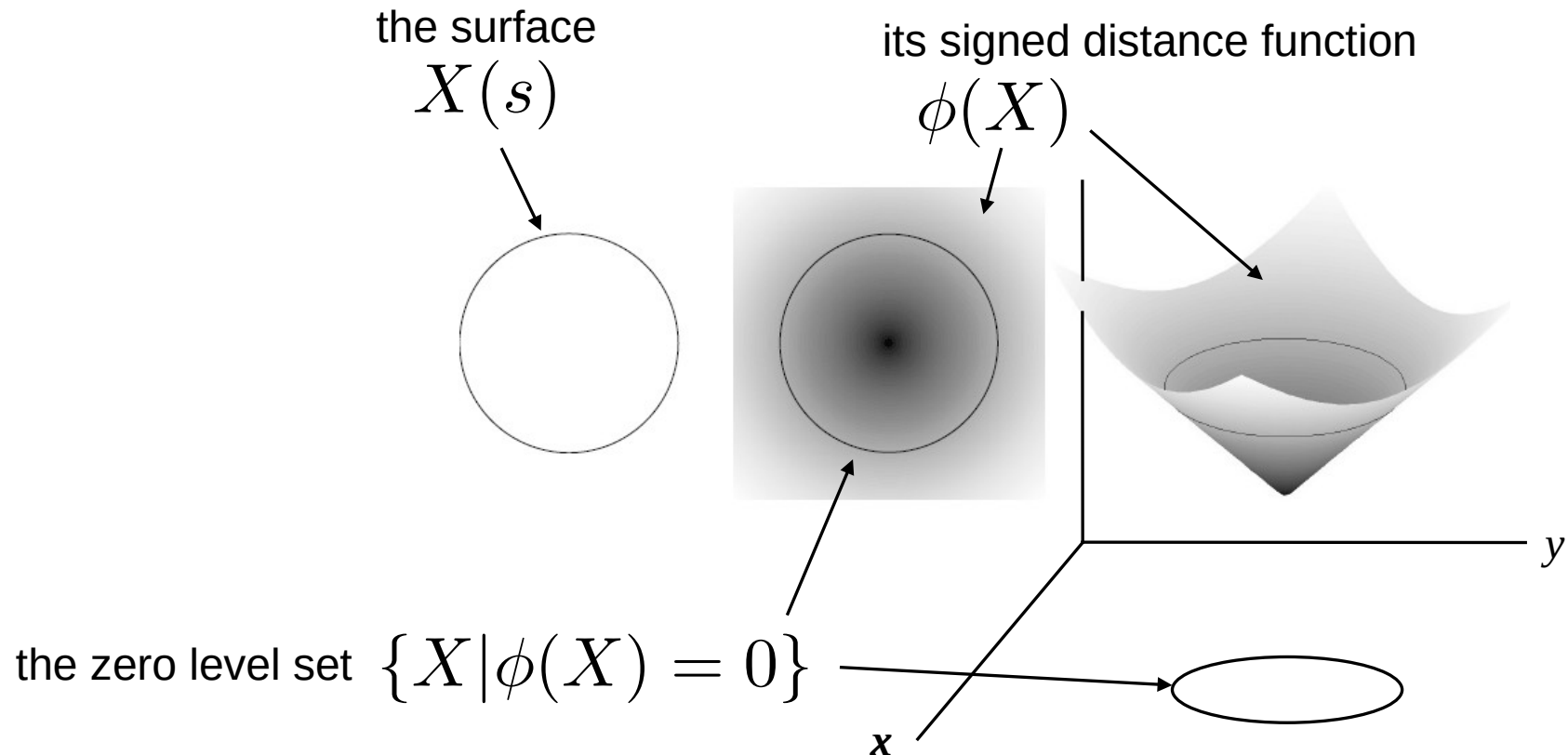
# Important concepts: level set surfaces

Surface:  $X(u, v) = [x(u, v), y(u, v), z(u, v)]$ ,  $u, v \in [0, 1]$

*Parametric model:* Piecewise-linear approximation:  
Points + triangles (meshes)



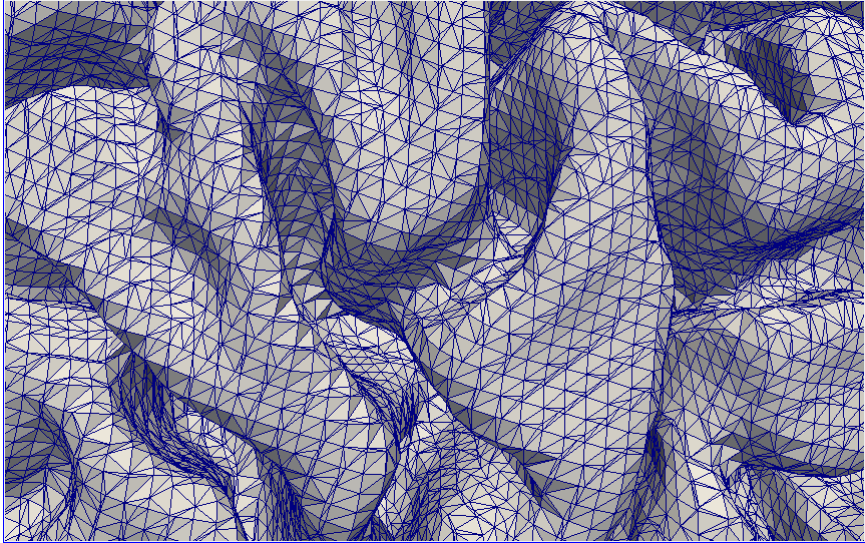
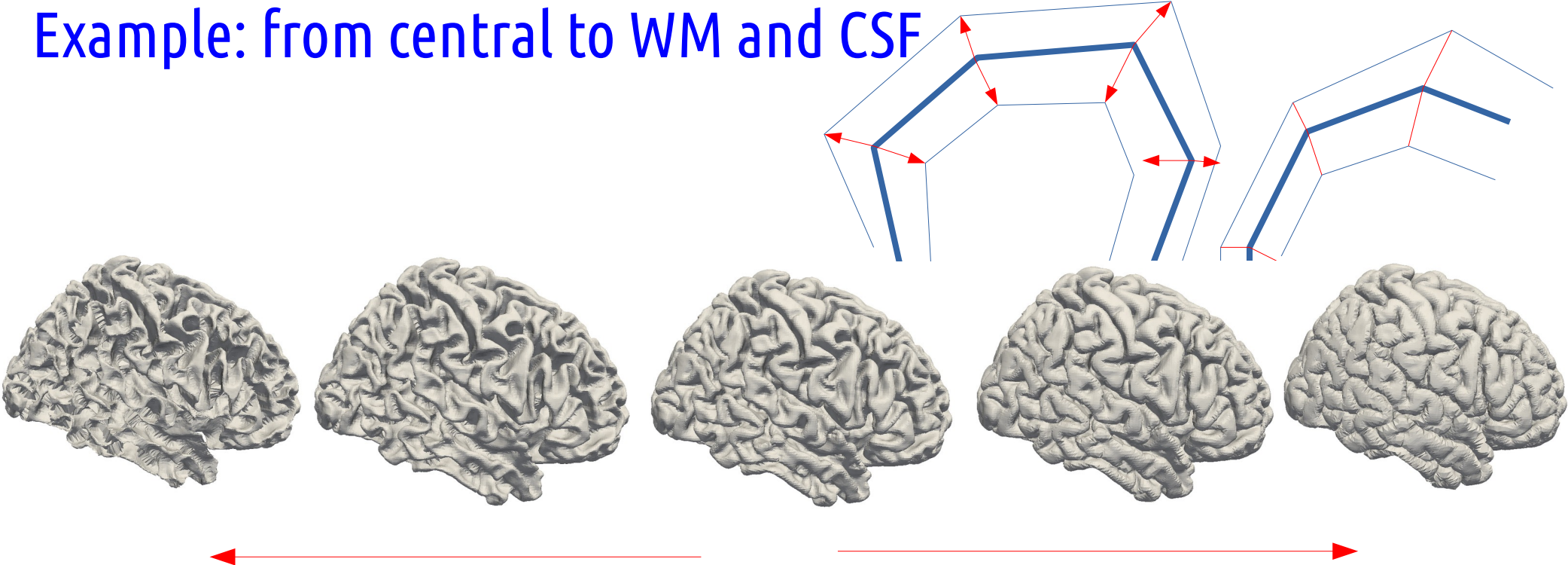
*Geometric model:*  $\phi(X(u, v)) = 0$





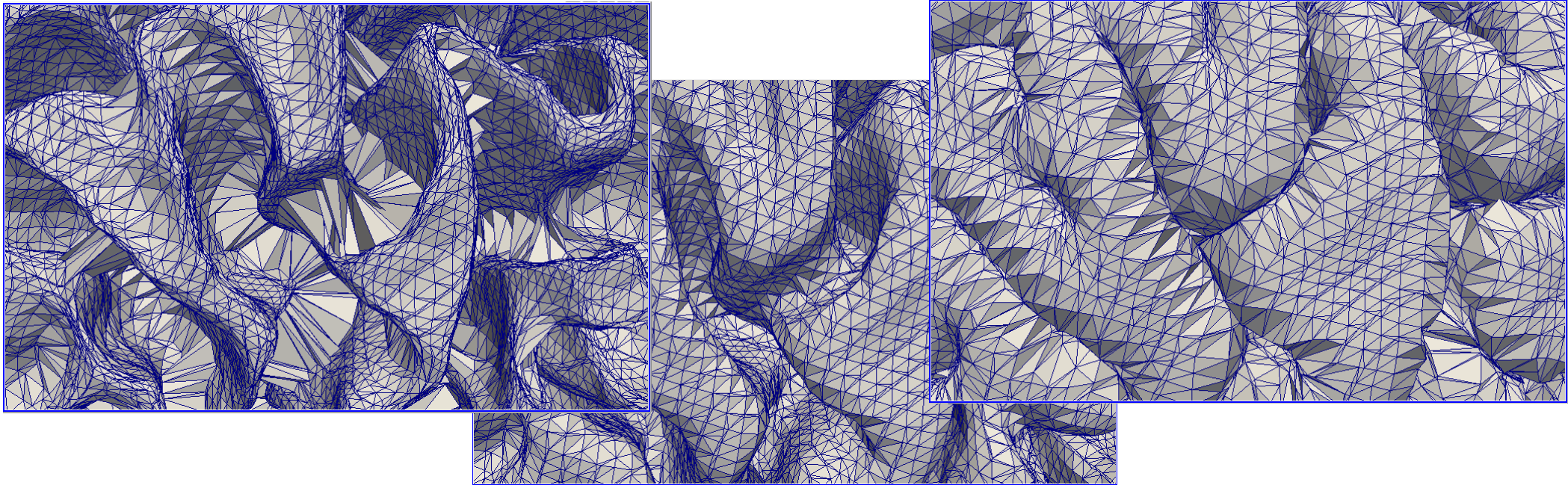
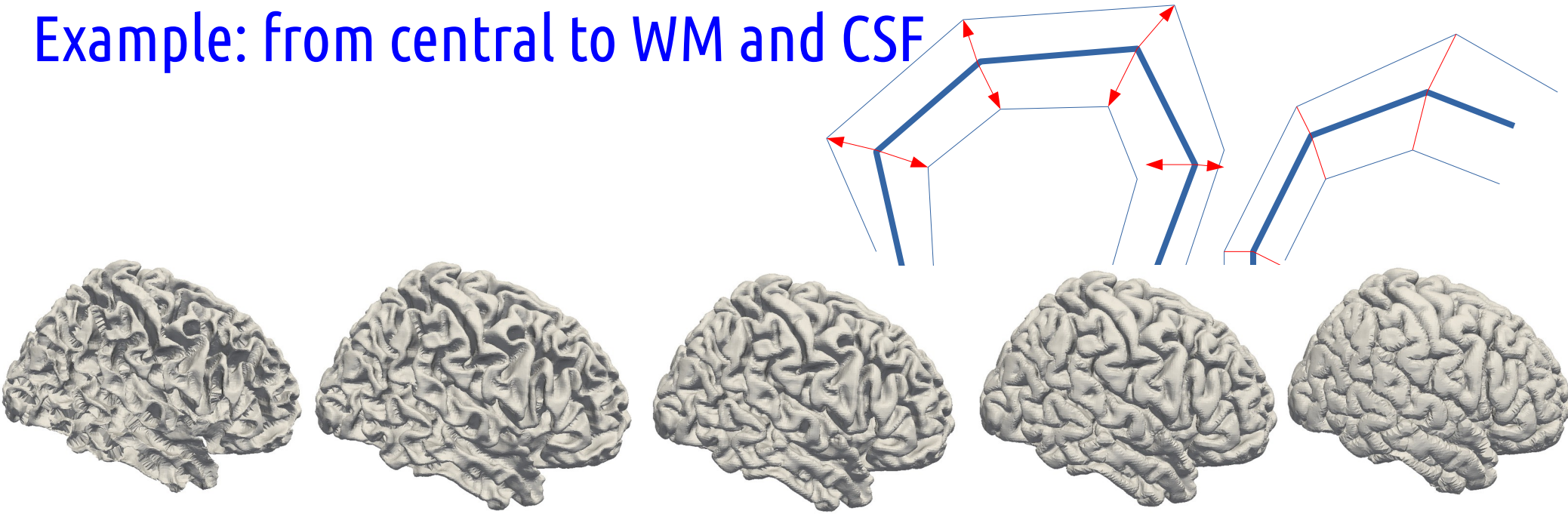


# Example: from central to WM and CSF

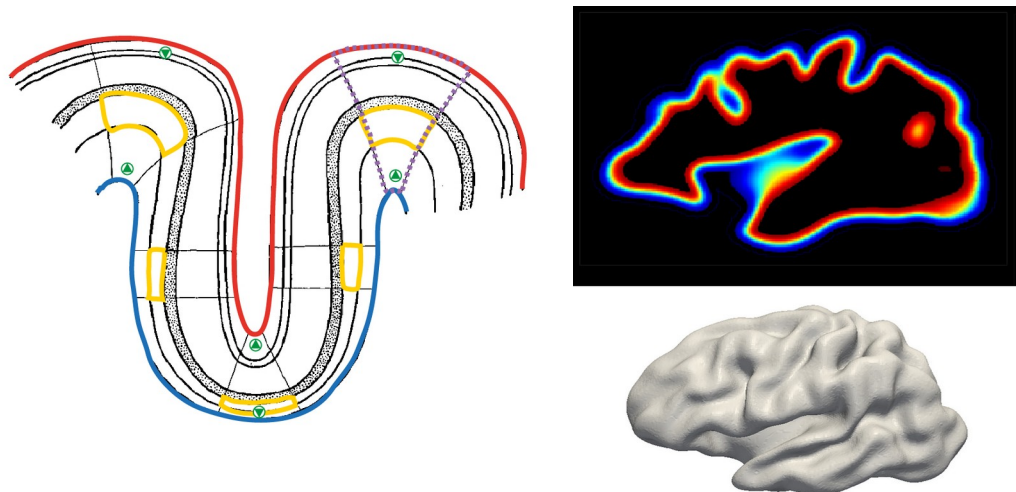




# Example: from central to WM and CSF



# From surfaces to laminar depth



Level sets of signed distance function

Defined in volumetric space

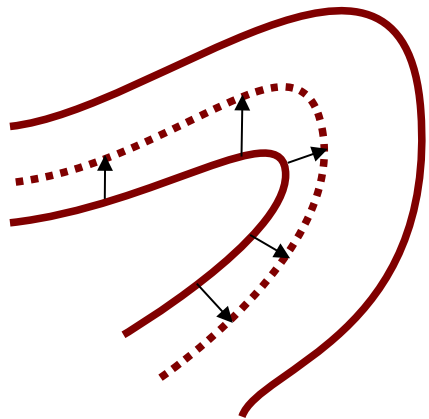
Constant sampling resolution

1-to-1 correspondence? With profiles

Surface processing? With lamination

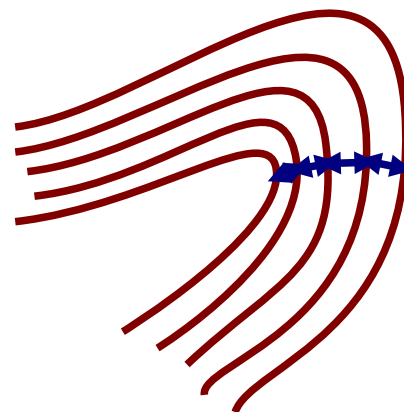
## *Dual lamination and profile estimation*

Level set lamination:

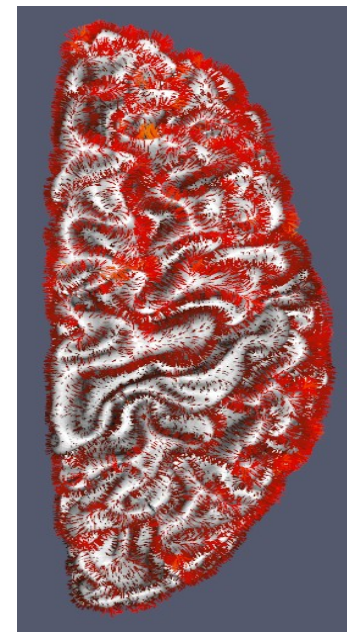
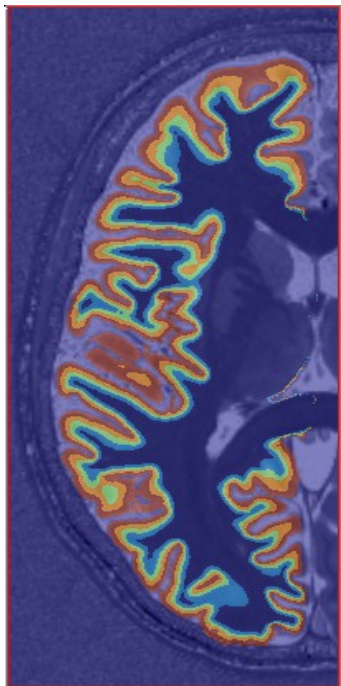


$$\frac{\partial \phi}{\partial t} = \alpha \kappa |\nabla \phi| + \beta (\phi - \phi_0) |\nabla \phi|$$

Cortical profile sampling:

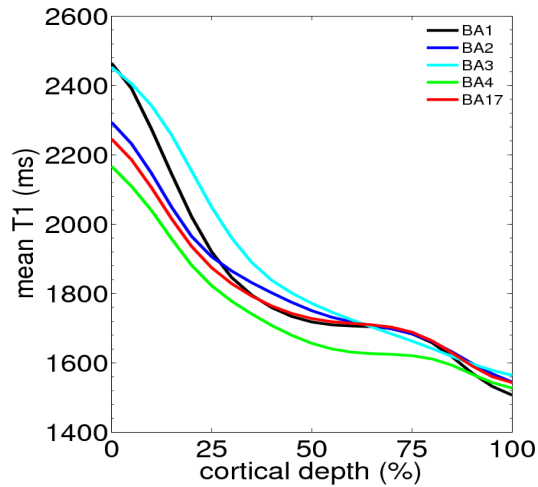


$$P_{l+1} = P_l - \phi_{l+1}(P_l) \frac{\nabla \phi_{l+1}}{\|\nabla \phi_{l+1}\|}(P_l)$$



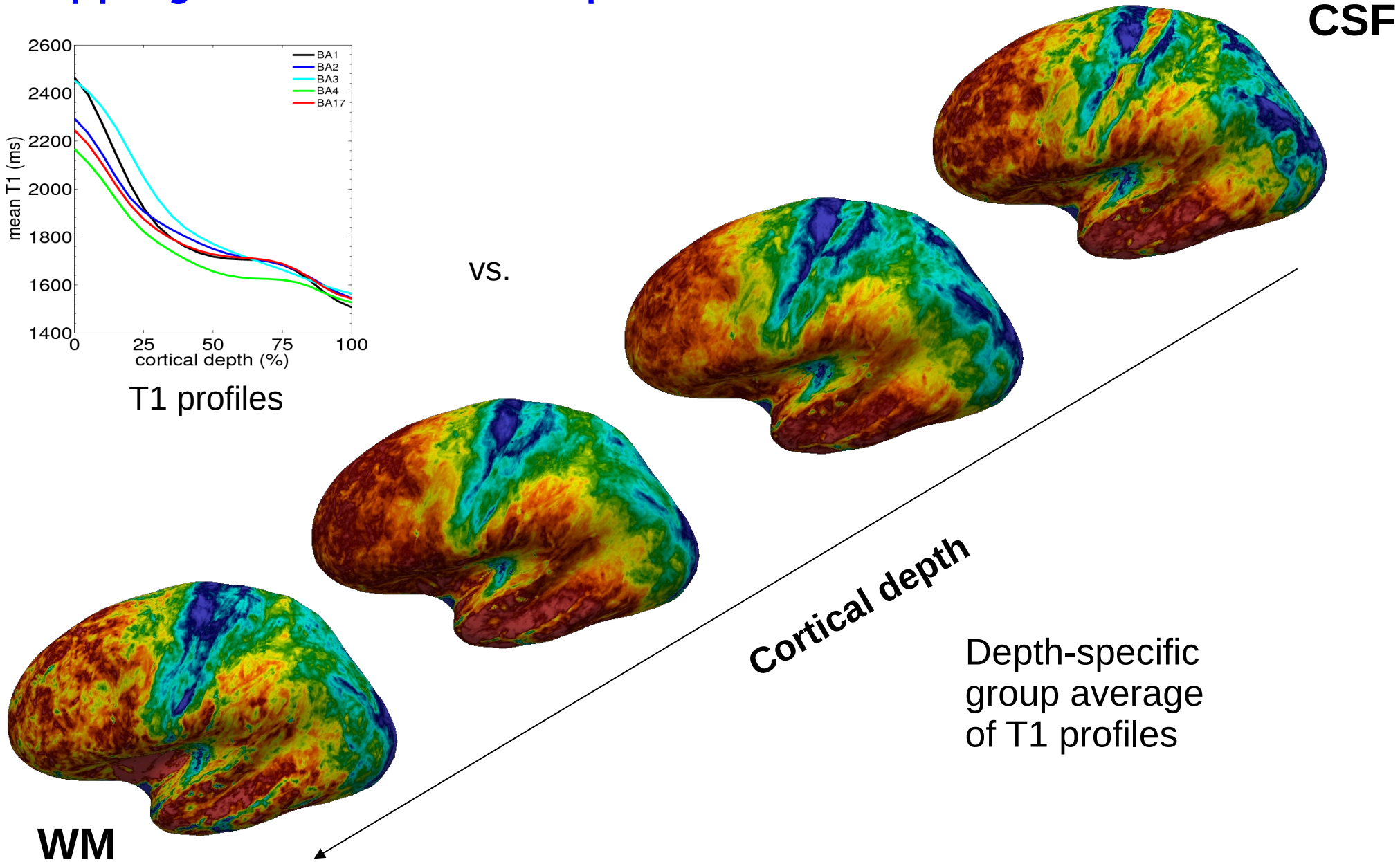


# Mapping across laminar depth



T1 profiles

VS.



WM

CSF

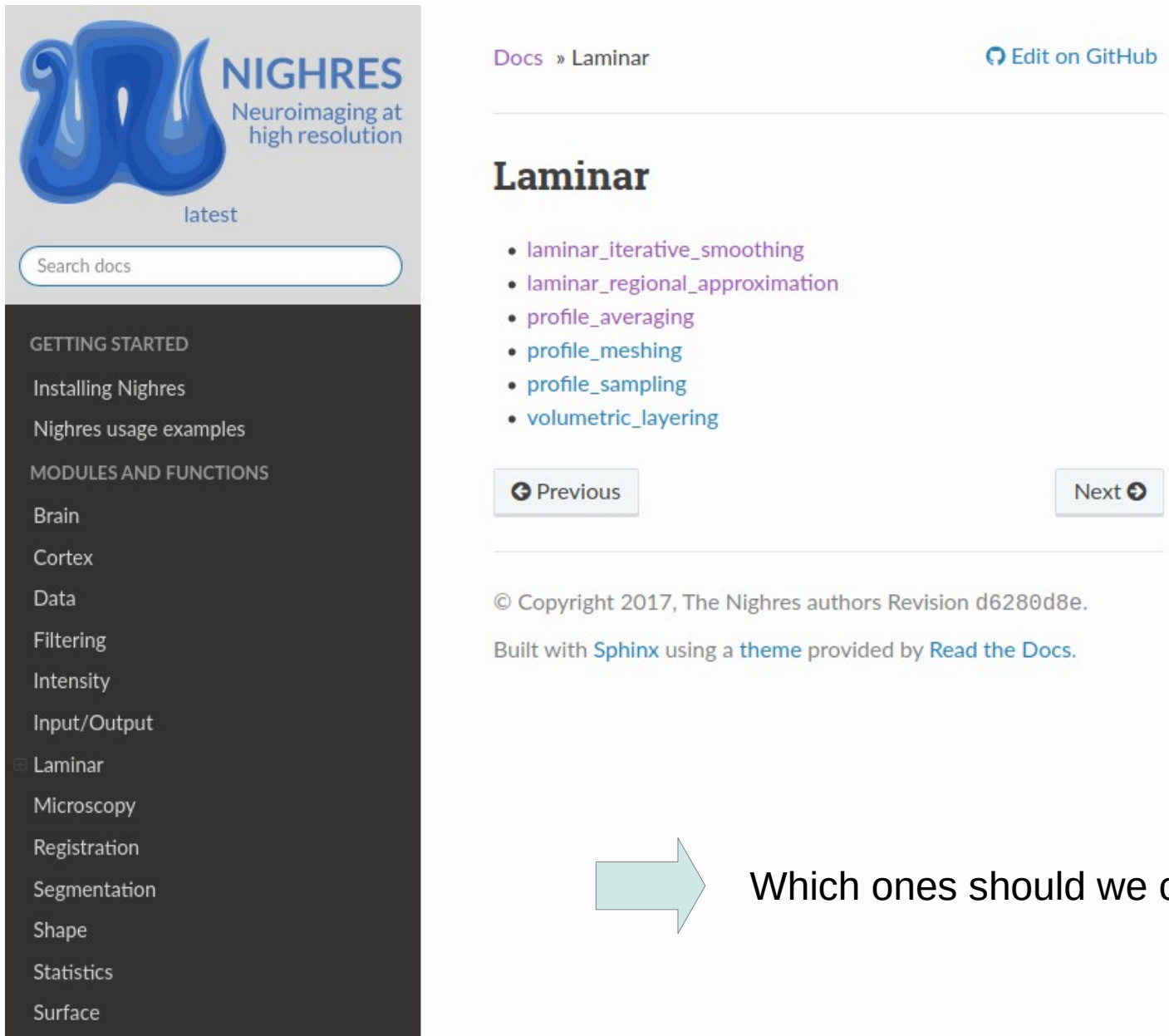
Cortical depth

Depth-specific group average of T1 profiles

Note: underlying mesh is the same

[Tardif et al., 2015]

# Nighres modules case by case



**NIGHRES**  
Neuroimaging at high resolution

latest

Search docs

GETTING STARTED

- Installing Nighres
- Nighres usage examples

MODULES AND FUNCTIONS

- Brain
- Cortex
- Data
- Filtering
- Intensity
- Input/Output
- Laminar**
- Microscopy
- Registration
- Segmentation
- Shape
- Statistics
- Surface

Docs » Laminar [Edit on GitHub](#)

## Laminar

- [laminar\\_iterative\\_smoothing](#)
- [laminar\\_regional\\_approximation](#)
- [profile\\_averaging](#)
- [profile\\_meshing](#)
- [profile\\_sampling](#)
- [volumetric\\_layering](#)

[Previous](#) [Next](#)

© Copyright 2017, The Nighres authors Revision d6280d8e.  
Built with [Sphinx](#) using a [theme](#) provided by [Read the Docs](#).

➔ Which ones should we cover?