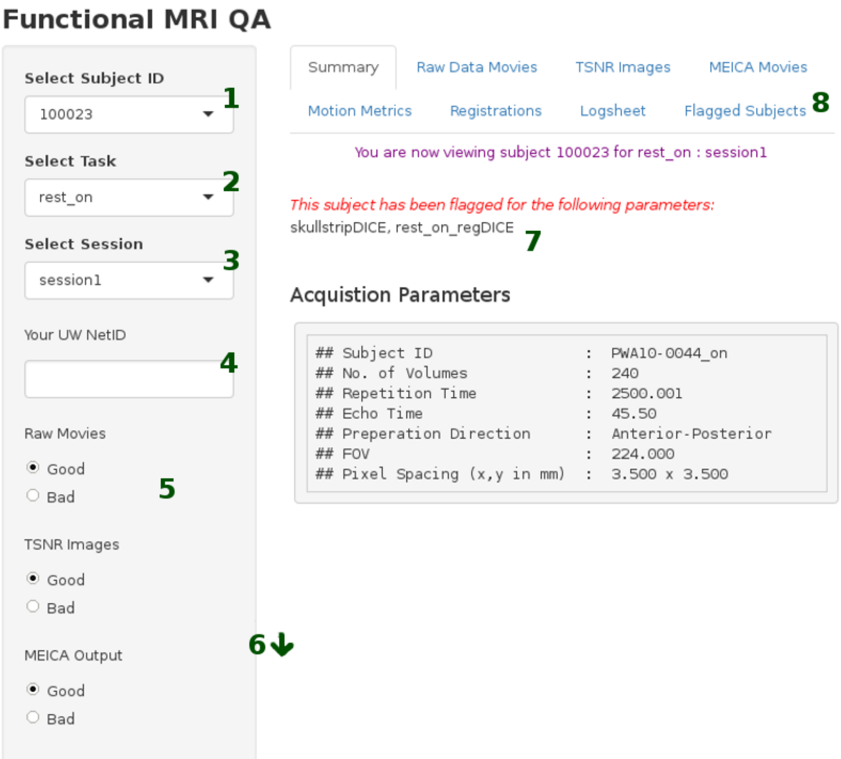
The IBIC Shiny QA app

1 Overview

For the Udall project, brain scan quality assurance procedures will be primarily completed using an R Shiny application, as a convenient and visually appealing alternative to command-line-based methods. Through this guide, we will cover how the app works, how to distinguish between "good" and "bad" images, and how to add warning checks.

Here's what the app looks like when it's first opened, and how it works:



Functional MRI QA

Select Subject ID: 100023

Select Task: rest_on

Select Session: session1

Your UW NetID:

Raw Movies: ● Good ○ Bad

TSNR Images: ● Good ○ Bad

MEICA Output: ● Good ○ Bad

You are now viewing subject 100023 for rest_on : session1

This subject has been flagged for the following parameters:
skullstripDICE, rest_on_regDICE

Acquistion Parameters

```
## Subject ID                  :  PWA10-0044_on
## No. of Volumes              :  240
## Repetition Time             :  2500.001
## Echo Time                   :  45.50
## Preperation Direction       :  Anterior-Posterior
## FOV                         :  224.000
## Pixel Spacing (x,y in mm)   :  3.500 x 3.500
```

1. **Subject ID**: All the scanned subjects are listed here. Choose different items on the drop-down menu to switch between subjects.
2. **Tasks**: Every PD subject will have four different tasks to be checked. Control subjects will have only the "on" tasks.
3. **Session**: Currently, all subjects have only session1; as the study progresses, more sessions will be added.
4. **UW NetID**: Input your NetID here to submit a QA report.

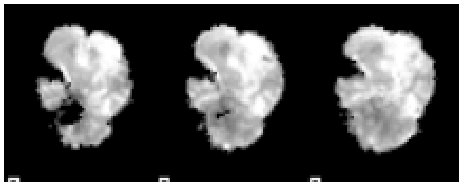
5. **Good/Fair/Bad**: For each measure, find the corresponding tab and select "good", "bad", or "fair" based on quality of the data.
6. **Submit a log**: Scroll down to the bottom of the sidebar, and click "submit". If you accidentally submit an incorrect log, go to ShinyQA/output/ and remove your most recent entry.
7. **Warnings**: If warnings show up, pay especially close attention to the scans they mention. These images are more likely to be problematic.
8. **Flagged Subjects**: These are the subjects that have warnings.

Next we will take a look at how to distinguish between "good" and "bad" images.

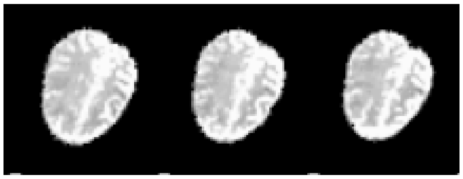
2 Checking Brain Scan Quality

Here are some examples of bad and good images. The bad images are unusable as they are. However, in some cases procedures can be taken to rectify certain issues.

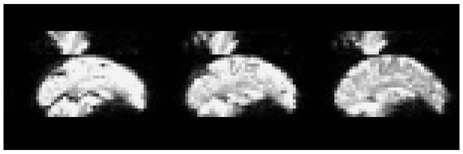
fMRI Movies:



Significant chunks of the brain are omitted, because the scanner has failed to pick up all of the brain.
The fix: none

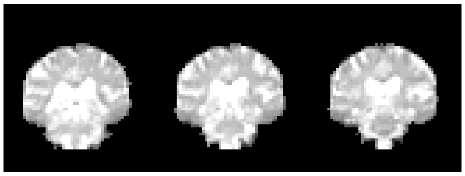


The brain is significantly tilted.
The fix: PROC



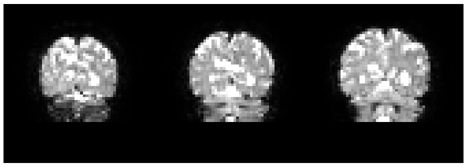
A movie has unexpected scrolling movements.
The fix: Scrap two volumes from the beginning and end of the movie, then merge frames for full images.

This is a usable image. No artifacts are present, and the whole brain is represented.

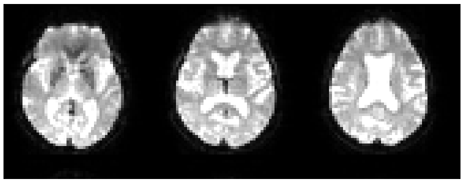


TSNR:
Bad TSNR images may appear blurred. Contrast may be incorrectly adjusted so that distinct brain structures are indistinguishable.

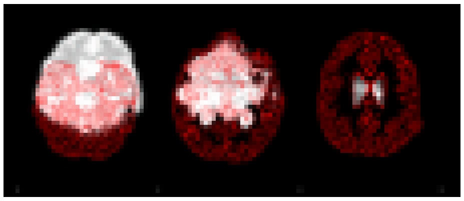
Brain structures are clear, and there is no obvious blurring. This is a good image.



This is a good image. Contrast here is good, with clarity in regards to brain structures.

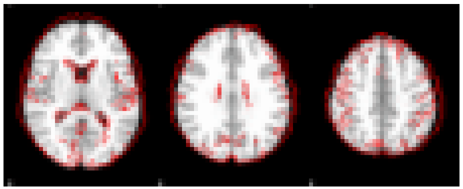


Registration:

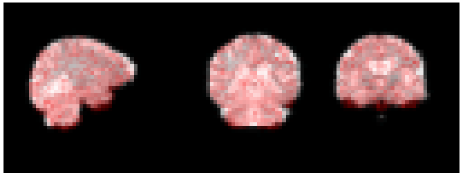


Registration appears to barely try matching the actual brain.
The fix: Re-run the registration processing. If the tracing is still terribly off, PROC

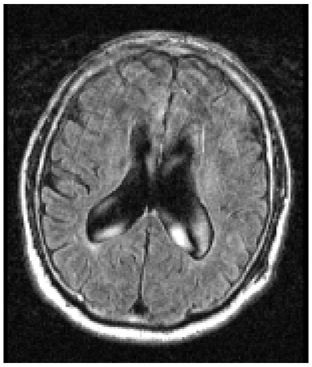
Registration looks *about* right. This image is acceptable.



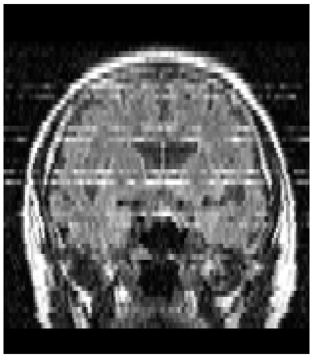
Registration follows the brain's contours fairly accurately. This image is good.



Other issues to look out for:



T1 images: Some ringing and blurriness can be perceived in the brain. This is due to motion during the T1 scan. No fix is currently available.



T1 images: Noise from the scanner itself appear on the image. Nothing can be done about these static-y artifacts, so this image is unusable.

For even more examples of what can go wrong with brain images, see this.

3 Warnings

Skullstrip DICE:
e.g. skullstripDICE, rest_on_regDICE
These warnings indicate that the process of stripping the skull away from T1 images might not have been entirely successful. Pay especially close attention to skullstripped images during QA.

Scan parameters:
e.g. RS_off_slices, ME_Task_on
These warnings indicate that the scan parameters for this subject's scan were incorrectly set.

Motion Metrics:
e.g. axcpt_on_dvarsvals_e002_mean, abs_mean_displacement, percent_outliers
Warnings generated for motion metrics are primarily an indication that something may be wrong with task-relevant brain images due to too much motion; in movies, look for excessive motion. In still images, look for blurriness.

4 Updating/Adding Warnings to the App

Each subject's warnings appear near the top of their summary page. Additionally, warnings for all subjects can be observed in the flagged subjects tab. These warnings are read from the file `QA/SUBJECTID_QA_stats.csv` in each subject's directory where `SUBJECTID` is variable. The file is structured as follows:

```
measure,data,flag
RS_off_slices,37,0
RS_off_echoes,3,0
RS_off_dyn_scans,240,0
RS_off_FOV_RL,224.000,0
...
```

and so on. When a flag = 1, the corresponding measure's name is displayed in the app as a warning. The data column contains the measure's quantitative value.

To generate these .csv files, run `make QA/SUBJECTID_QA_stats.csv` in each subject's directory. Alternatively, one can also call the `generate_QA_stats` script directly with two arguments `SUBJECTID` and `SESSIONNO` where the latter is the desired session number. Here is the default location of the QA csv-generating script:

```
/mnt/panuc/IBIC_Pipelines/ShinyQA/generate_QA_stats
```

Altering flag thresholds: All thresholds should be numerical values that are global fields that can be found in the beginning of the script. To change the thresholds, simply edit these existing fields and regenerate all subjects' csv files after the update.

Adding potential warnings: Follow the output format (measure,data,flag) as shown above when adding to the script `generate_QA_stats`. This script generates flags in two stages, for task-independent and then task-dependent measures. Measures are task-dependent if they can be categorized as either `rest_on`, `rest_off`, `axcpt_on`, or `axcpt_off`. Add flag computations under the appropriate stage. Then re-generate the csvs for all subjects under the new script.

Future Improvements: Whenever a QA stats csv file is updated, it is regenerated from scratch. As the flags being generated are currently inexpensive to compute, regeneration is quick. However, if more time-consuming flags are to be computed in the future, it would be a good idea to look into caching such flags in such a way that they need not be recomputed.