Tutorial to Run MRIQC Batch Processing Using Docker and Bash

This tutorial explains how to automate running MRIQC — a tool for MRI quality control — on multiple subjects using a Bash script that executes Docker containers for each subject.

Prerequisites

- **Docker installed and running** on your system.
- Data organized in **BIDS format** with folders named like `sub-XXX` for each subject.
- Permissions to read the data directories and write to output and work directories.
- A Linux or macOS environment to run the script.

Directory Structure

Variable	Description	Example (anonymized)
`rawdata_dir`	Directory containing the BIDS data (read-only)	`/path/to/bids_data`
`output_dir`	Directory where MRIQC outputs will be saved	`/path/to/mriqc_output`
`work_dir`	Directory for temporary working files	`/path/to/workdir`

Bash Script

```
#!/bin/bash

# Directories — replace with your actual paths
rawdata_dir="/path/to/bids_data"
output_dir="/path/to/mriqc_output"
work_dir="/path/to/workdir"
```

```
# Loop over each subject directory
for dir in "$rawdata_dir"/sub-*; do
    # Extract the subject ID from the folder name (e.g., sub-01)
    subject_id=$(basename "$dir")

    # Run MRIQC Docker container for the current subject
    docker run -it --rm \
      -v "$rawdata_dir":/data:ro \
      -v "$output_dir":/out \
      -v "$work_dir":/scratch \
      nipreps/mriqc:latest /data /out participant \
      --participant_label "$subject_id" \
      -w /scratch --write-graph --notrack --ants-float
done
```

Step-by-step Usage

1. **Update the paths**:
    - Change the variables `rawdata_dir`, `output_dir`, and `work_dir` to your actual directories.
2. **Permissions**:
    - Make sure you have read access to your BIDS data (`rawdata_dir`).
    - Ensure you have write permissions on `output_dir` and `work_dir`.
3. **Save the script**:
    - Save the above script into a file, e.g., `run_mriqc.sh`.
4. **Monitor the processing**:
    - The script will run MRIQC on each subject folder (`sub-*`).
    - Results will be saved in `output_dir`.
    - Temporary files and logs will be stored in `work_dir`.

Explanation of Key Script Components

- `for dir in "$rawdata_dir"/sub-*; do ... done`
  Iterates through all subject folders starting with `sub-`.

- `subject_id=$(basename "$dir")`
  Extracts just the folder name, e.g., `sub-01`.

- `docker run -it --rm \ ... nipreps/mriqc:latest ...`
  Runs the MRIQC Docker container, mounting directories:
  - `-v "$rawdata_dir":/data:ro`: mounts input data as read-only.
  - `-v "$output_dir":/out`: output directory.
  - `-v "$work_dir":/scratch`: scratch directory for temporary files.

- MRIQC command-line options:
  - `/data /out participant`: input, output, and participant-level run mode.
  - `--participant_label "$subject_id"`: processes only this subject.
  - `-w /scratch`: working directory inside container.
  - `--write-graph`: writes workflow graph.
  - `--notrack`: disables anonymous usage tracking.
  - `--ants-float`: uses floating point precision for faster ANTs processing.

Additional Tips

- To run subjects in parallel, consider using GNU Parallel or background jobs.
- Check MRIQC documentation for more options and details.
- Make sure you have enough disk space in your `work_dir` for temporary processing files.