|  |
| --- |
| **A close-up of a label  Description automatically generatedFigure 1.**  *Cicada Pipeline* |

**WORKING WITH CICADA**

The Comprehensive Independent Component Analysis Denoising Assistant (CICADA) is a novel ICA-based denoising method applicable for both resting-state and task-based fMRI data. CICADA is designed for after preprocessing (including normalization to MNI space but without smoothing) but before statistical analyses. CICADA uses manual IC classification guidelines1 (the gold-standard of ICA denoising) to help identify all commonly established sources of fMRI noise. Specifically, CICADA uses the IC spatial maps, timeseries, and power spectrums to classify ICs as signal or various types of noise. CICADA can be split into three main different sections. First is Automatic CICADA, which performs subject-level ICA denoising and quality control automatically. Second is Manual CICADA, a fully optional step, which allows a user to perform a highly efficient version of manual IC denoising with quality control analyses. Third is Group CICADA, which will perform group-level processing and quality control of the CICADA-denoised data. Altogether, CICADA aims to effectively automate the manual gold standard, make the gold-standard more efficient and approachable, and significantly ease quality control analyses following denoising.

**CICADA Use Cases**

CICADA is currently only compatible with data warped to the MNI 2009c asym adult template. Therefore, it is not currently compatible with young pediatric data or data that cannot be warped to MNI 2009c asym space. This will hopefully be updated later. CICADA works well with fMRIPrep preprocessing but fMRIPrep is not required. CICADA is compatible with resting-state and task data. See “How to Run Automatic CICADA” for more details.

If updates to CICADA are made, this will be detailed on the Github (<https://github.com/keithcdodd/CICADA>).

**CICADA Installation**

The most recent version of CICADA is available for download from the author’s Github page (https://github.com/keithcdodd/CICADA). To install and use, a user needs to download the CICADA script folder. The user must also have FSL and Matlab installed and working. Likely necessary Matlab Add-Ons include Statistics and Machine Learning Toolbox, Image Processing Toolbox, Image Acquisition Toolbox, and Bioinformatics Toolbox. More Matlab toolboxes (“Add-ons”) may be necessary.

CICADA Dependencies:

* FSL (tested on version 6.0.5.2)
* Matlab (tested on version 2022a)
  + Statistics & Machine Learning Toolbox, Image Processing Toolbox, Image Acquisition Toolbox, Bioinformatics Toolbox

FSL must be set up to work with gzipped nifti files (.nii.gz). For example, FSLOUTPUTTYPE must be ‘NIFTI\_GZ’ and FSL must be able to connect to Matlab (e.g., path must include fsl/etc/matlab). Similarly, the system path ($PATH variable on mac/unix, PATH environmental variable on Windows) must have FSL in it. When run, the Auto\_CICADA script will attempt to fix these variables for the user automatically. However, the correct set up may differ depending on the user, operating system configuration, or updates. The current methods can be referenced in the figure below (from the Auto\_CICADA function) and adjusted to fit the needs of the user:

|  |
| --- |
| A screenshot of a computer code  Description automatically generated  **Figure 2.** *FSL & Matlab Compatibility Set Up* |

**How to Run Automatic CICADA**

Broadly, Automatic CICADA is carried out through three scripts (referred to as “base scripts”): 1. CICADA\_1\_MasksandICAs.sh, 2. CICADA\_2\_AutoLabeling.m, and 3. CICADA\_3\_QC.m. They are to be run in order for each fMRI scan. These scripts can be run either individually or can be automatically called by the wrapper script Auto\_CICADA.m. Notably, if the data was preprocessed via fMRIPrep, this wrapper script will be called as part of fmriprep\_auto\_CICADA.m.

**Running Automatic CICADA following fMRIPrep preprocessing:**

CICADA requires BIDS filenames and directory structure format (https:// https://github.com/bids-standard), as shown in the figure below. If the data was preprocessed using fMRIPrep (e.g., version 20.2.6), then the output files will already be in BIDS format.

|  |
| --- |
| **Figure 3.** *Necessary Directory Formatting (BIDS)* |

CICADA can typically be run on fMRIPrep processed data with the fmriprep\_auto\_CICADA.m command. However, the folder/file/naming format can differ depending on the version of fMRIPrep and the structure of the underlying data. Therefore, it is possible that instead of using the fmriprep\_auto\_CICADA.m command, a user should instead use the command (see section). If the fmriprep\_auto\_CICADA.m command is used, it should be applied to each folder for each image. For example, at the Matlab command prompt:

> fmriprep\_auto\_CICADA(/path/to/fmriprep/data/folder, /path/to/cicada/output/folder, sub\_id, ses\_id, task\_name, anat\_ses\_id, redo\_mel, mel\_fol, task\_events\_file, compare\_file, tolerance)

Input argument “sub\_id” is the subject ID number, “ses\_id” is the session ID number, “task\_name” is the name of the task, “anat\_ses\_id” is the ID number for the session where the best/preferred anatomical scan is from fMRIPrep (often times, this is either the same as the ses\_id, or just the first session [“01”]). In the case that there is no best session where the anatomical scan is held (i.e., the anat folder is not held within a session folder), a user can either give an empty “anat\_ses\_id” (anat\_ses\_id = ‘’), or just set it to the same as the ‘ses\_id’. The “redo\_mel” numerical input should normally be 1, to always run melodic to generate ICs, regardless of if it has been run in the past.

Input “task\_events\_file” is the path to a task events file that details the task onsets (if it is a task design). If it is resting state, or a user would rather not include a task events file, this parameter can be left as an empty character string: ‘’. The task events file, if used, is often named similar to “task-foodpics\_run-01\_events.tsv” and must be in the following format:

|  |
| --- |
| A table of numbers with black text  Description automatically generated  **Figure 4.** *Task Events File Format* |

Specifically, there must be onset, duration, and trial\_type columns. In the example given above, the task was a block-design foodpics task. If a task events file is provided, CICADA will use it to generate individually estimated HRF responses for each trial type and one combined estimated HRF response, to help capture ICs that are related to each trial type. Thus, if this can be applicable to a user’s task-design, it is highly suggested to be included. Task data can be block or event-related design. If the task data is an event-related design, the duration (in seconds) would have to be adjusted accordingly (e.g., duration of 1 TR in seconds could be seen as relatively equivalent to an impulse or event). Specifically, the onset and durations columns are used by CICADA to create a box function to convolve with the HRF response.

The “compare\_file” is the fMRI data to be compared to the CICADA-denoised data. By default, the compare file is the 8 parameter (8p) regression denoised image created by CICADA. Finally, the input argument “tolerance” is the tolerance value to use during “ CICADA\_2\_AutoLabeling.m”. In short, tolerance relates to the maximum number of signal-probability-sorted ICs in a row that CICADA labels as noise before automatically labeling all other ICs also as noise. For more details on tolerance, see details in “Basescript 2: CICADA\_2\_AutoLabeling.m” section. The default tolerance value (5) is suggested.

In general, “mel\_fol”, “task\_events\_file”, “compare\_file”, and “tolerance” values are not necessary to run the fmriprep\_auto\_CICADAfunction (especially when the data is resting state with no “task\_events”\_file).

For example, running CICADA on sub-102, ses-01, task-rest, and fmriprep folder located at /home/fmriprep, output directory /home/cicada, the Matlab command would be:

> fmriprep\_auto\_CICADA('/home/fmriprep', '/home/cicada', '102', '01', 'rest', '01', 1)

**Running Automatic CICADA for Preprocessed Data With or Without fMRIPrep:**

If the data was preprocessed outside of fMRIPrep, or the fMRIPrep folder/file structure is different than given above, the necessary inputs can always be provided through the Auto\_CICADA.m function in the wrappers folder. The command looks like this:

> Auto\_CICADA(output\_dir, funcfile, funcmask, confoundsfile, redo\_mel, mel\_fol, compare\_file, task\_events\_file, anatfile, anatmask, gm\_prob, wm\_prob, csf\_prob, tolerance)

Where “output\_dir” is the desired CICADA output directory, the “funcfile” is the functional file and “funcmask” is the funcmask. The “confoundsfile” input is normally, by default, the fMRIPrep confounds file. This file is usually named similar to “sub-102\_ses-01\_task-rest\_desc-confounds\_timeseries.tsv”. In the case where fMRIPrep was not used, a confounds file that is in the same format as the fMRIPrep file (e.g., .tsv filename) needs to be created. It should look like the following:

|  |
| --- |
| A table of numbers and lines  Description automatically generated  **Figure 5.** *Confounds File Format* |

The confounds file can contain many more confounds columns (like the fMRIPrep file), but it must have the confounds listed here, with the same column names. Altogether, it must have the nine common parameters (six motion, white\_matter, csf, global\_signal), as well as dvars, framewise\_displacement, and rmsd.

The “redo\_mel”, “mel\_fol”, “compare\_file”, “task\_events\_file”, and “tolerance” inputs are the same as mentioned in the “Running Automatic CICADA following fMRIPrep preprocessing” section.

The “anatfile” input is the anatomy file, usually a T1. If not provided, the MNI 2009c asym T1 will be used. The “anatmask” input is the anatomy mask. The “GM\_prob”, “WM\_prob”, and “CSF\_prob” inputs are the probability files for gray matter, white matter, and cerebral spinal fluid. If these three probability files are not provided, CICADA will default to the standard MNI 2009c asym probability files.

**Automatic CICADA Methods**

CICADA methods are all contained within the basescripts which are called by Auto\_CICADA.m. The methods used by each basescript are described in detail below.

**Basescript 1: CICADA\_1\_MasksandICAs.sh**

Broadly, the first basescript creates anatomical and functional masks, performs FSL’s MELODIC to generate ICs, and provides relevant calculations of these ICs, using the previously created masks, to prepare for IC classification in the second basescript.

Many new masks are generated by the first basescript. First, a new functional mask is created to ensure that areas outside of the brain tissue (but generally excluding the skull) are included. This helps ensure that CICADA will be able to characterize both sinus flow and edge artifact. This is accomplished by first taking the maximum of the original functional mask and the anatomy mask (resampled to functional space). Second, the output from the first step is masked both by a 6 mm gaussian smoothed anatomical mask and a lightly thresholded unmasked functional file to form the final functional mask. Altogether, this creates a functional mask large enough to include sinus flow and edge artifact, but without retaining large amounts of other material (e.g., skull, eyes, etc.). The unmasked functional file is then masked by this new functional mask (which is used as the functional mask for all future calculations as well).

Several region-based probability files and masks are then created. In each case, a probability file (an approximation of the relative chance that a voxel belongs to the given region) is created first. Then, a mask of this region is generated, usually by thresholding at a robust 67th percentile. First, a stringent susceptibility probability and mask is calculated. In short, functional file voxels whose values fall at or above the robust 50th percentile are given a susceptibility probability of 0% with susceptibility values increasing up to 100% as functional file voxels decrease to the 0th percentile. Next, the edge region is calculated. First, the susceptibility mask is subtracted from the functional mask. Second, an eroded and smoothed functional mask is subtracted to give the edge region. Gray matter (GM), white matter (WM), and cerebral spinal fluid (CSF) regions are then calculated. First, their corresponding probability files (which can be provided from the user but will default to the ones from fMRIPrep) are resampled, then the edge and susceptibility probability files are subtracted and then masked by the functional mask. Next, a subependymal region is calculated. In short, first, CSF and WM are masked by an eroded anatomy mask. The GM probability is then subtracted out and then smoothing is applied. The overlap between these modified CSF and WM files are then used to estimate the subependymal region. Next, a NotGM region is calculated by subtracting the GM region from the functional mask. An “inbrain” region is generated by subtracting the subependymal region from a combined GM and WM region. An “outbrain” region is then formed from subtracting this “inbrain” region from the functional mask. Finally, an “outbrain\_only” region (a region containing mostly outside sinuses and not the inner CSF) is created by subtracting the CSF, subependymal, edge, susceptibility, and eroded anatomy mask regions from the outbrain region. This concludes the major region/mask generations of the first basescript. An example image of the most relevant region masks is detailed in the CICADA manuscript, but also provided here.

|  |
| --- |
| A close-up of a brain  Description automatically generated  **Figure 6.** *CICADA-Generated Noise Profile Regions Example* |

After creating the relevant masks, the first basescript then performs FSLs MELODIC to generate the ICs. Next, estimates for IC smoothness and IC spatial map probabilities are calculated. Probabilities are calculated by merging and thresholding probability maps from MELODIC. Two different IC spatial maps are generated to later calculate smoothness. First, a non-thresholded IC z-stat map is created from MELODIC outputs and absolute valued. Second, the non-thresholded IC z-stat map is smoothed (6 mm gaussian) and absolute valued. The second basescript later uses these two maps to calculate a smoothness parameter. In short, the smoother the IC spatial map data is, the less data is “lost” to 6 mm gaussian smoothing (e.g., positive signal is “less cancelled out” by negative signal). The first basescript then also resamples a brain network template (from Guzmán-Vélez et al.2, and adapted from Yeo et al.3) to the functional space.

The final part of basescript one involves calculating the IC spatial map overlap with each relevant mask generated near the start of this script (alongside the networks from the adult brain network template, if desired). This outputs the overlap mean and number of voxels. This then sets up for basescript two, which will calculate the total overlap for each relevant region from these values.

**Basescript 2: CICADA\_2\_AutoLabeling.m**

Broadly, the second basescript classifies the ICs and applies nonaggressive denoising. First, a smaller, more constrained, functional mask is generated. The purpose of this constrained mask is to help later generate an improved group functional mask. Next, an HRF response is estimated using a double gamma HRF. If the functional file is a task-based scan, and a task events file is provided, an HRF response estimate based on the task events file is also generated. Next, the second basescript uses the outputs of the first basescript to calculate the general IC spatial overlap with each relevant region and network. The relative proportion of each region overlap compared to the total spatial IC map is also calculated. Smoothness (“smoothing retention”) is estimated by dividing the summation of the non-thresholded IC z-stat map by the smoothed one (see “Basescript 1: CICADA\_1\_MasksandICAs.sh” section). Next, general power frequency proportions are calculated and categorized by low frequency (<0.008 Hz), BOLD (0.008-0.15 Hz), and high frequency (>0.15 Hz). The overlap between the IC power frequencies and the estimated HRF responses power frequencies is also calculated (“hrf power frequency overlap”). Next, timeseries correlations to DVARS, Framewise Displacement (FD, the version proposed by Power 20124), the six motion parameters, white matter, CSF, and global signal, are all calculated. These parameters are pulled from a user-provided confound file but will default to pull from the fMRIPrep confound file if it exists. Specifically, a detrended DVARS and FD are correlated to a detrended, differentiated, and absolute valued timeseries. This method is chosen as DVARS and FD are, by definition, differentiated and absolute valued parameters as well. General IC timeseries “spikiness” is also estimated by the maximum absolute value of the normalized timeseries.

The second basescript then uses k-means clustering, into 3 groups, to cycle through relevant noise profiles (see manuscript for more detail on noise profiles), brain networks, and other related variables detailed above. The initial k-means starting points are given as the minimum, median, and maximum values. Thus, the ICs are clustered as either high, medium, or low in that noise profile, network, or other variable. Many of these classifications are labeled as either “good” (highly likely to indicate neural signal) or “bad” (highly likely to indicate noise). Basescript two then re-sorts the ICs, from high to low, based on the following equation:

NSP is the “relative Neural Signal Probability”, SR is “smoothing retention”, GMO is “Gray Matter spatial map Overlap”, and HRFO is “HRF power frequency Overlap.” In each case, “norm” refers to a normalization of each parameter to values ranging from [0,1]. Overall, this equation takes advantage of the fact that neural signal is characterized by ICs with higher smoothness, higher gray matter overlap, and higher power frequency overlap with the estimated HRF response. GMO is squared to greater weigh its value, as gray matter overlap is likely more specific of neural signal than either smoothness or HRF power overlap. This can be inferred, for example, by the fact that certain noise profiles (e.g., Subependymal) can be highly smooth and have great HRF power overlap but not have high GM overlap.

After re-sorting the ICs from high to low based on NSP, CICADA loops through each IC to classify each IC as either (neural) signal, or noise. For an IC to labeled as signal, the IC needs to meet the following criteria:

IC Signal Labeling Criteria:

1. The IC needs to be k-means classified as high in GMO, HRFO, or SR.
2. The IC either needs high GMO or needs no other classified “bad region” to be in the high k-means group.
3. If the IC is high in GMO and high in HRFO, this is enough to mark as signal regardless of other potential negatives.
4. If there are no “bad” labels (e.g., no high bad classifications) mark the IC as signal, given that signal labeling criteria 1 is met.

While looping through each IC in NSP order, CICADA uses a tolerance value (default is 5, but can be modified by the user) to determine when to stop. Whenever CICADA labels an IC as noise, the tolerance value is reduced by one. Similarly, when CICADA labels an IC as signal, the tolerance value is increased by one (but is never raised above the starting value). If the tolerance value reaches 0, CICADA stops looping through the ICs and labels the rest of the ICs as noise. CICADA will also not loop through ICs whose NSP is less than the mean IC NSP. Therefore, any IC whose NSP is less than the mean NSP will also be labeled as noise.

If less than two ICs are labeled as signal initially, CICADA will correct this by labeling the highest two NSPs as signal. This acts as a failsafe to the code structure. Later, Group CICADA (see “Group CICADA Methods” section) will also label each image that has less than 3 ICs labeled as signal as outliers to not be used.

Next, the second basescript will create an excel spreadsheet (“IC\_auto\_checker.csv”) that details the selection process. The “IC\_auto\_checker.csv” includes the IC number, what CICADA labeled it as (signal or noise), if the IC was high in signal or noise, any good (signal-like) or bad (noise-like) classifications given to the IC, and if the IC may have been grouped toward a particular brain network. An example image of part of one of these files is given here. Of note, the “SignalLabel” column contains CICADA’s final decisions on IC classifications where a 1 is signal, and a 0 is noise.

|  |
| --- |
| A screenshot of a computer  Description automatically generated  **Figure 7.** CICADA *IC\_auto\_checker Example* |

Finally, basescript two performs a few last actions to make it easier for a user to examine the classification and output. CICADA creates structures to hold all relevant calculations, values, and classifications. CICADA also creates a “compare cleaning” table to compare the relative feature proportions of all ICs to just the ICs that CICADA labeled as signal. CICADA also generates a signal-to-noise ratio image and generates images containing all noise and all signal IC overlaps. Files to perform nonaggressive denoising (the standard default method) and aggressive denoising are saved and exported. Then, basescript two performs nonaggressive denoising via fsl\_regfilt, alongside 8 parameter regression (six motion parameters + mean WM + mean CSF) and 9 parameter regression (8 parameter + mean global signal) for comparison. All relevant variables are saved and exported.

**Basescript 3: CICADA\_3\_QC.m**

Basescript three generates useful quality control (QC) analyses for each fMRI image. This assists in quality control analyses and in evaluating how well CICADA performed. This also helps inform a user if the IC selection for an image should be adjusted and rerun (“Manual CICADA”).

In creating QC plots, basescript three will default to comparing the CICADA-denoised file to the 8-parameter-denoised file, both generated by basescript two. The user, however, can provide the function with a different denoised file to compare to (e.g., 9 parameter denoising). First, basescript three estimates the final temporal degrees of freedom and outputs the percent of variance that is retained in the data following CICADA denoising. Basescript 3 then calculates the relative noise profile correlations (e.g., correlations within the edge region, correlations within CSF, correlations between GM and framewise displacement, etc. ). Finally, basescript three plots the noise profile correlations of both the CICADA denoised data and the comparison denoising data for reference. This concludes Automatic CICADA.

**How to Run Manual CICADA**

To perform CICADA-assisted Manual IC denoising (“Manual CICADA”), or to adjust the labeling of Automatic CICADA for an image (e.g., upon QC inspection CICADA did not label the data accurately enough), the user should do the following general steps:

1. Perform Manual IC Classification by adjusting the “IC\_auto\_checker.csv” signal labels
2. Resave the adjusted file as “IC\_manual\_checker.csv”

Performing manual IC classification in CICADA is designed to be an efficient process following the implementation of Automatic CICADA. First, the user should open the “IC\_auto\_checker.csv” created by Automatic CICADA. The user should then perform manual IC classification by adjusting the “SignalLabel” column, where a 1 is a signal label and a 0 is a noise label (following manual IC denoising guidelines1). Next, the user should resave the file with the new name “IC\_manual\_checker.csv”. After this is accomplished, the user can run Manual CICADA. This is done in a similar manner (with similar inputs) to Automatic CICADA. Instead of fmriprep\_auto\_CICADA.m, use fmriprep\_manual\_CICADA.m. Instead of Auto\_CICADA.m, use Manual\_CICADA.m.

Altogether, Manual\_CICADA calls CICADA\_2\_Manuallabeling.m and then CICADA\_3\_QC.m again on the manually-adjusted data. CICADA\_2\_Manuallabeling.m simply reads in the “IC\_manual\_checker.csv” file for the new labels, re-does the nonaggressive denoising, and re-outputs and saves new quality control variables and files. The inputs and actions of these scripts are otherwise similar to the Automated versions. Manual CICADA will not rerun the first basescript.

**Manual CICADA Methods**

Manual CICADA methods are similar to Automatic CICADA. In short, Manual CICADA will use a user-adjusted “IC\_auto\_checker.csv” file (renamed as “IC\_manual\_checker.csv”) to perform manual IC denoising. This is accomplished with a greatly shortened basescript two (which reperforms nonaggressive denoising) and the same basescript three as in Automatic CICADA.

**How to Run Group CICADA**

After running all of either Automated or Manual CICADA on each individual fMRI image of interest, Group CICADA should be performed. In short, the purpose of Group CICADA is to run quality control analyses on the denoised data and reorganize the data and the quality control information in an easily accessible and understandable manner. Altogether, Group CICADA should be run on one task (“task\_name”) at a time but can include any number of subjects/sessions.

To perform Group CICADA, a user should run the cicada\_group\_qc.m function:

> cicada\_group\_qc(cicada\_home, group\_qc\_home, task\_name, output\_dirname, file\_tag, smoothing\_kernel, fpass, redo\_melodic, sub\_ids, ses\_ids, excludes, outliers, adjusteds, task\_event\_files)

The “cicada\_home” input is the CICADA output directory used in the Automatic/Manual CICADA (where the CICADA denoised data is held). The “group\_qc\_home” input is the desired parent output directory where the results of Group CICADA will be output. The “task\_name” input is the same as in previous functions. The “output dirname” is the output folder name for the current image group/type. Altogether, when running cicada\_group\_qc.m on a set of image data, the output directory will be in “group\_qc\_home/task\_name/output\_dirname”. The “file\_tag” input should be a string that is unique to the file naming of the type of denoised file in Group CICADA. For example, if running Group QC on Automatic CICADA, the “file\_tag” could be ‘\_auto\_’, or for Manual CICADA ‘\_manual\_’, or for 8 parameter regression ‘\_8p\_’.

Automatic and Manual CICADA do not apply smoothing or bandpass filtering. Instead, filtering and then smoothing can be applied at the group level near the end of Group CICADA. The “smoothing\_kernel” is the gaussian smoothing kernel FWHMx size (default smoothing size is equivalent to the original functional resolution and can be automatically applied with smoothing\_kernel = -1;). The “fpass” input is the Hz for bandpassing given as an array from low to high (if used, fpass = [0.008, 0.15]; is recommended). Note, QC plotting following Group CICADA is easiest to interpret with minimal/no smoothing and no bandpass filtering. Smoothing, and potentially bandpass filtering, will create new within-noise-profile-region correlations in the QC plots. CICADA denoising also already decreases low frequency (<0.008 Hz) and high frequency (>0.15 Hz) noise without bandpass filtering, and, regardless, detrending to the 2nd polynomial is performed. If performing ROI analyses, no smoothing (smoothing\_kernel = 0;) or bandpass filtering (fpass = [];) is recommended. Otherwise, minimal smoothing (from the default value and up to 6 mm), as well as no bandpass filtering, is recommended, but remains up to the user.

Group CICADA will also run MELODIC on the group level. The “redo\_melodic” input, then, is the same as in previous functions, but for the group-level MELODIC. The “sub\_ids” and “ses\_ids” inputs are similar to previous functions (referring to the subject ID and session ID) but should be cell arrays of equal length that specify each subject/session pair for the task in the group analysis. For example, if the group analysis was on sub-102 session 01, sub-102 session 02, sub-103 session 01, and sub-108 session 02 for Group CICADA, the variables would be set as follows: sub\_ids = {‘102’, ‘102’, ‘103’, ‘108’}; and ses\_ids = {‘01’, ‘02’, ‘01’, ‘02’};. The “task\_event\_files” parameter is also the same as in previous functions, but again as a cell array of the same length as sub\_ids and ses\_ids. This is optional, but like in previous functions, a good idea for task data.

Finally, “excludes”, “outliers”, and “adjusteds” are all parameters that can tell the script to handle specific data differently than the rest. All three parameters should be cell arrays the same length as “sub\_ids” and “ses\_ids” and should be a ‘1’ if they belong in the given group (“excludes”, “outliers”, or “adjusteds”). Specifically, “excludes” is for data that the user does not want included in the Group QC analysis. This might apply, for example, to data where there was a scanner error, and the resulting image was found to be unusable. In the example from the previous paragraph, for example, perhaps MRIQC software identified sub-108 session 02 to be an extreme outlier in data quality from all other data (e.g., because of a scanner error). Then we would set the excludes variable to excludes = {‘0’, ‘0’, ‘0’, ‘1’}; which would then ensure that sub-108 session 02 is not included.

“Outliers”, while similar, is a bit different from “excludes”. The “outliers” parameter is for data that was not previously (before IC generation and CICADA processing) found to be necessary to exclude. However, upon MELODIC IC generation, or CICADA processing, it was found to give significant issues that make it undeniably obvious it is not saved by IC denoising. For example, perhaps sub-103 session 01 did not generate any ICs that truly look like signal, then this data could be marked as an outlier with outliers = {‘0’, ‘0’, ‘1’, ‘0’}; . Functionally, the main difference between “excludes” and “outliers” is whether the data will be included in the group CICADA folder. Excluded images will be entirely skipped by Group CICADA whereas outliers will still be copied into the group output with its QC parameters included. Outliers, however, will be marked as such to help a user perform future analyses without the given image if desired.

Finally, the “adjusteds” parameter is for data that a user performed Manual CICADA on (they adjusted the signal labels of Automatic CICADA, and then ran Manual CICADA). The “adjusted” parameter could be used if, when a user evaluates the individual and Group QC, an image is found to be an outlier but Automatic CICADA did not label the ICs well. Then, a user could adjust the signal labeling and run Manual CICADA solely for that data and then mark it as an “adjusteds”. This ensures that Group CICADA correctly pulls the manually adjusted data for that image. For example, let’s say in our example that for sub-102 session 02, the QC plots show poor results but are greatly improved upon performing Manual CICADA. Then, a user could consider including the manually-adjusted data in the Group CICADA analyses, and mark it as an adjusteds like so: adjusteds = {‘0’, ‘1’, ‘0’, ‘0’}; .

**Group CICADA Methods**

Group CICADA is performed through the cicada\_group\_qc.m function.

**Group CICADA: Cicada\_group\_qc.m**

Broadly, Cicada\_group\_qc.m performs group level adjustments, QC analyses, and prepares the data for statistical analyses. First, the function copies over all individual QC comparison plots originally generated by Automatic/Manual CICADA to one folder. This allows for easy QC plot comparison across images. Additionally, Cicada\_group\_qc.m copies over and combines all the individual data together into a single folder. For each image, the function uses user-defined inputs to determine if the current image should be entirely excluded or if Manual CICADA data (if it exists) should be used instead of Automatic CICADA. Cicada\_group\_qc.m also similarly pulls the original data for each image (before denoising) and the 8p denoised image for QC comparison. Next, the specified detrending, bandpass filtering, and smoothing of the CICADA, 8p, and original data is performed.

Following the processing of CICADA, 8p, and the original data, the relevant QC information relevant for group QC for each of the three images are calculated. This is accomplished in several steps. First, commonly used QC cut-off values are calculated. For example, Group CICADA calculates the median and mean FD values, the %FD > 0.2 mm, if there are any FD values > 5 mm, the median DVARS, and the mean RMSD. Next, correlations within each CICADA noise profile (Edge, FD, DVARS, Outbrain, WMCSF, CSF, NotGM, Suscept) and within GM are calculated in the same manner as CICADA\_3\_QC.m. Altogether, Group CICADA continues to calculate potentially relevant QC values and images. This includes, but is not limited to, a ratio of the gray matter mean temporal variance divided by the “NotGM” (area outside of GM) temporal variance, a signal and noise IC spatial overlap image, what proportion of gray matter is covered by signal ICs, what proportion of signal ICs is found within the gray matter, a dice coefficient of gray matter and signal IC overlap, and the number and percent of ICs labeled as signal. All relevant QC calculations are then combined into relevant tables.

From there, Group CICADA performs a few last steps to aid in QC analyses. In short, this involves concatenating the QC data for each image together and calculating potential data outliers. QC measures are marked as outliers if they are three scaled median absolute deviations from the median (“MAD outlier”). Altogether, Group CICADA automatically labels an image as a Group CICADA outlier if the image is a MAD outlier in any of the following:

Group CICADA Outlier Identification:

1. Low gray matter coverage by the signal ICs
2. Low gray matter to signal IC dice value
3. Low ratio of GM to NotGM mean variance
4. Low power overlap with the HRFO
5. Low BOLD frequency to High Frequency ratio
6. Low number of ICs labeled as signal
7. < 3 ICs labeled as signal

Group CICADA will also label conservative outliers (where either the mean\_FD > 0.25 mm, the %FD > 0.2mm is > 20%, or any FD > 5 mm) and liberal outliers (mean\_FD > 0.55 mm). The cut-offs for conservative and liberal outliers are adapted from Satterthwaite et al. 20135. Next, Group CICADA saves a group qc table (containing all calculated QC data for all images, including the different types of outliers [conservative, liberal, CICADA]), a group QC correlation table (containing the sampled noise profile correlations), and a Group QC plot (same as the ones from CICADA\_3\_QC.m but for the whole group instead of per image). Finally, group MELODIC is run to generate group-level ICs. The resulting ICs that best match each of the seven networks (Medial Visual, Sensory Motor, Dorsal Attention, Ventral Attention, FrontoParietal, Default Mode, Subcortical) from the brain network image are also returned. This is determined, in short, by sorting each IC by their dice coefficient to each network, and then testing, in order, if including the IC overall increases the resulting total dice coefficient. Altogether, this could help a user better evaluate the success of the denoising in capturing different common brain networks. This concludes Group CICADA. Following QC analysis, a user could, for example, use the image\_names.txt in the Group CICADA folder to select the CICADA-denoised data and perform statistical analyses.

**References**

1. Griffanti, L. *et al.* Hand classification of fMRI ICA noise components. *Neuroimage* **154**, 188–205 (2017).

2. Guzmán-Vélez, E. *et al.* Amyloid-β and tau pathologies relate to distinctive brain dysconnectomics in preclinical autosomal-dominant Alzheimer’s disease. *Proc. Natl. Acad. Sci. U. S. A.* **119**, e2113641119 (2022).

3. Thomas Yeo, B. T. *et al.* The organization of the human cerebral cortex estimated by intrinsic functional connectivity. *J. Neurophysiol.* **106**, 1125–1165 (2011).

4. Power, J. D., Barnes, K. A., Snyder, A. Z., Schlaggar, B. L. & Petersen, S. E. Spurious but systematic correlations in functional connectivity MRI networks arise from subject motion. *NeuroImage* **59**, 2142–2154 (2012).

5. Satterthwaite, T. D. *et al.* An improved framework for confound regression and filtering for control of motion artifact in the preprocessing of resting-state functional connectivity data. *Neuroimage* **64**, 240–256 (2013).