Supplementary Materials

Definitions

Batch effects refer to systematic occurrence of a technical artifact within a subset of a cohort[2]. For example, the slice spacing may be different between two institutions, creating a batch effect if those studies were evaluated in unison. This could result in Site A producing MR volumes with a larger slice spacing as well as a majority of the cases belonging to the positive target class, while Site B provides MR volumes with much smaller slice spacing and a majority of cases from the negative class. When combining data from both sites, any developed model must be optimized to identify biologically relevant features specific to the target class, as opposed to determining that slice spacing may be relevant for distinguishing between the 2 classes (as the latter is really an acquisition difference between the 2 sites). Figure S-1 visualizes such distinctive variations in The Cancer Genome Atlas Glioblastoma Multiforme (TCGA-GBM) collection[1].

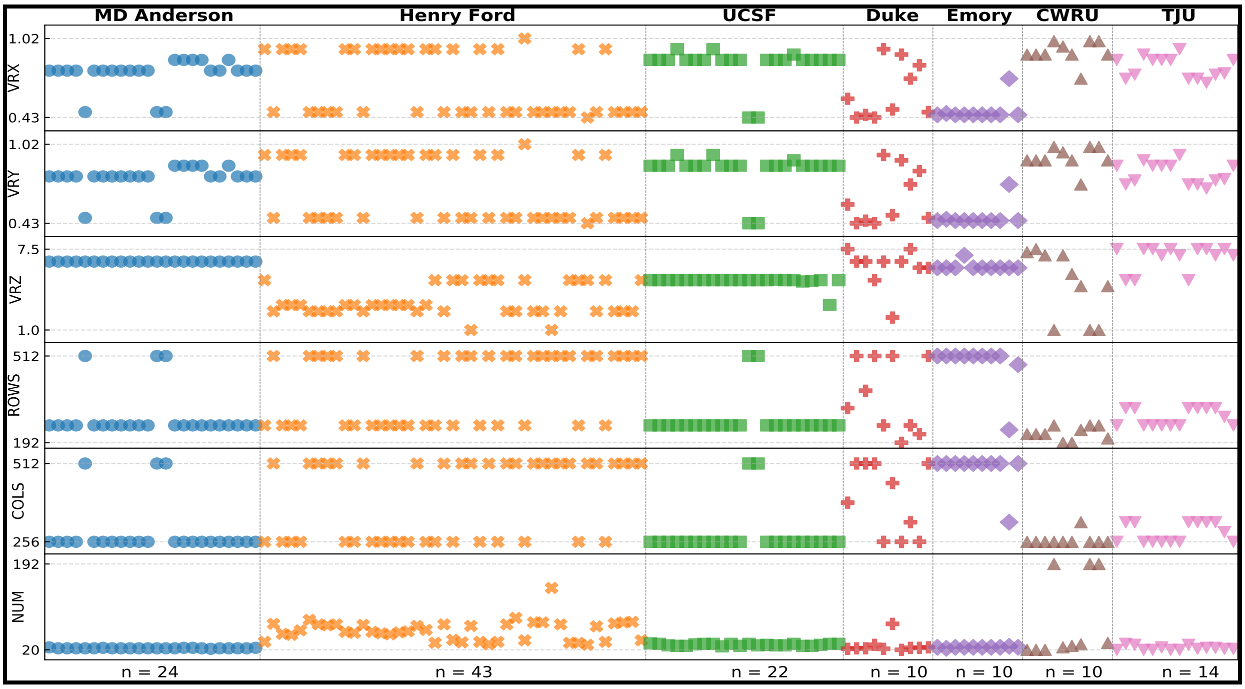

Figure S-1: Variations in MRI scan metadata illustrating batch effects within the TCGA-GBM cohort across 133 post-contrast T1-weighted MRI datasets curated from 7 different sites (in different colors) where each point corresponds to a unique subject. VRX, VRY, and VRZ correspond to voxel resolutions in-plane (x, y) and through-plane (z) respectively. $ROWS$ and $COLS$ represent in-plane scan size and the NUM is the number of images in each volume. All information was directly extracted from DICOM metadata of each MRI dataset, as downloaded from TCIA.

Image artifacts refer to the image quality issues such as the presence of noise, motion, shading, lack of detail, ringing, aliasing or other issues; that are present within individual MR datasets to varying degrees[3]. These typically occur as a result of the acquisition process, scanner calibration issues, or technician error.

Overview of the foreground detection algorithm

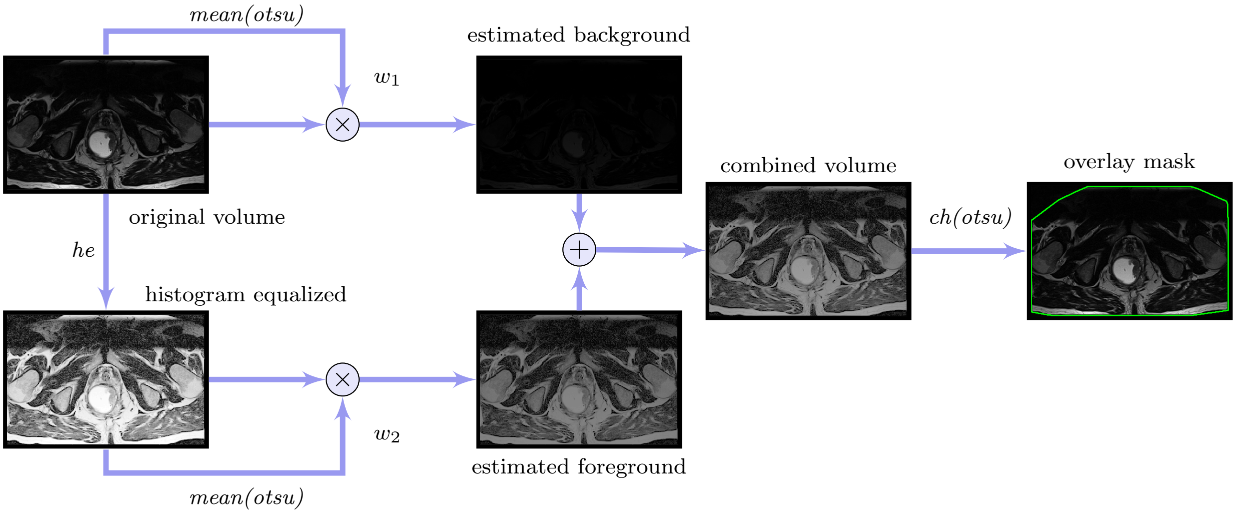

Figure S-2: Schematic of the foreground detection algorithm.

Figure S-2 shows the schematic of our proposed foreground detection algorithm. Otsu thresholding is typically proposed as an efficient approach to quickly identify both these regions in an MRI volume, but common shadowing/shading artifacts that are present in MRI volumes are known to significantly affect its performance[4]. By contrast, histogram equalization could help articulate details in the MRI volume even with shading or shadow artifacts, but is known to also intensify the appearance of noise[5]. In order to take advantage of both methods simultaneously, a weighted combination of the original MRI volume and its histogram equalized version were used as an input to Otsu thresholding. The weights were defined automatically to ensure a completely unsupervised method. A convex hull operation was then used to enclose the foreground based on the mask output from Otsu thresholding. While this algorithm was designed primarily for scans with one primary foreground region, it has been generalized to be able to detect and return multiple individual foreground objects that may be present in an MR image (e.g. axial slices over both legs[6]). The Supplementary Materials section has additional details on settings that can be specified by the end-user in this regard. These steps were implemented using the `scikit-image` python package[7].

Format and Usage

MRQy has been made publicly accessible as an open-source project through GitHub (https://github.com/ccipd/MRQy), and can be downloaded as well as contributed to freely by any end-user. Specific package dependencies for MRQy have included in the GitHub documentation. After the installation of all the prerequisite Python packages (specified in the installation instructions), MRQy can be run on a directory containing files for a given cohort via the command: `python QC.py output_folder_name "input directory address"`. No additional configuration files need to be specified. This results in the following steps being executed:

- Thumbnail images are generated for all 2D sections in each MRI dataset and saved as *.png* files within the `UserInterface/Data` folder.
- Each dataset is processed to detect the foreground and background region.
- *Metadata* are extracted from file headers for each dataset. *Measurements* are computed based on the detected foreground region for each dataset.
- Both metadata and measurements are saved for each dataset within a tab-separated file (*results.tsv*) that is stored within the `UserInterface/Data` folder.
- For a given cohort, a single UMAP and a single t-SNE embedding are computed for all the datasets based on the 23 measures (after whitening). The embedding co-ordinates are also saved into the *results.tsv* file.

Further interrogation of cohort variations and artifact trends may be done reading *results.tsv* into any common data analytic tool (e.g. MATLAB or R). A specialized front-end HTML interface (*index.html*) is available within the `UserInterface` folder designed for real-time manipulation and visualization. Quality control can be performed via multiple pathways:

- Using sorting arrows available on each table column to re-order measures and examine numeric trends. Users can further annotate rows or remove non-informative patients.
- As the different interface components are synchronized, if a patient row is highlighted in either Table, a corresponding highlight appears on a line within the PC chart, on a bar in the bar chart, as well as shading the patient-specific bubble in the embedding plots. Thumbnail images for this patient volume are shown in the interface.
- Using the PC and bar charts to directly compare a specific measure across all the subject scans. This can help quickly determine which of the metadata or measures are consistent across the entire cohort as well as identify outliers. The PC chart can also be used to evaluate positive or negative relationships between different measures[8] and thus determine the trade-off in processing for specific artifacts.
- Using embedding plots (t-SNE and UMAP) to track specific site- or scanner-specific trends within the cohort. By visualizing the 2D space into which the entire cohort has been mapped, any clusters that can be identified typically correspond to site- and scanner-specific variations. The overall distribution

of points in space also provide an indication of the variability within the entire cohort.

User-Specified Settings

By default, MRQy extracts a basic set of 10 metadata tags from *.dcm* files, summarized in Table 2. Additional tag fields or private metadata can also be extracted by specifying them via a *.txt* file and using the syntax:

```
python QC.py output_folder_name "input directory" -t "tags .txt address".
```

The user can also specify a configuration for the foreground detection algorithm using the following command:

```
python QC.py output_folder_name "input directory" -c "False".
```

Note that the default value for the `-c` flag is *False*, where MRQy measures are computed across all foreground objects together. When the `-c` flag is *True*, MRQy will identify each foreground object separately and compute a measurement per individual object.

References

[1] L. Scarpace, T. Mikkelsen, S. Cha, S. Rao, S. Tekchandani, D. Gutman, J. Saltz, B. Erickson, N. Pedano, A. Flanders, J. Barnholtz-Sloan, Q. Ostrom, D. Barboriak, and L. Pierce, Radiology Data from The Cancer Genome Atlas Glioblastoma Multiforme [TCGA-GBM] collection, The Cancer Imaging Archive , http://doi.org/10.7937/K9/TCIA.2016.RNYFUYE9.

[2] J. T. Leek, R. B. Scharpf, H. C. Bravo, D. Simcha, B. Langmead, W. E. Johnson, D. Geman, K. Baggerly, and R. A. Irizarry, Tackling the widespread and critical impact of batch effects in high-throughput data, Nature Reviews Genetics **11**, 733–739 (2010).

[3] D. C. Van Essen et al., The Human Connectome Project: A data acquisition perspective, NeuroImage **62**, 2222–2231 (2012).

[4] Y. Feng, H. Zhao, X. Li, X. Zhang, and H. Li, A multi-scale 3D Otsu thresholding algorithm for medical image segmentation, Digital Signal Processing **60**, 186 – 199 (2017).

[5] S. Navdeep, K. Lakhwinder, and S. Kuldeep, Histogram equalization techniques for enhancement of low radiance retinal images for early detection of diabetic retinopathy, Engineering Science and Technology **22**, 736 – 745 (2019).

[6] S. Z. K. Sajib, N. Katoch, H. J. Kim, O. I. Kwon, and E. J. Woo, Software Toolbox for Low-Frequency

Conductivity and Current Density Imaging Using MRI, IEEE Transactions on Biomedical Engineering **64**, 2505 – 2514 (2017).

[7] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors, scikit-image: image processing in Python, PeerJ **2**, e453 (2014).

[8] A. Inselberg, Multidimensional detective, in *Proceedings of VIZ '97: Visualization Conference, Information Visualization Symposium and Parallel Rendering Symposium*, pages 100–107, 1997.