Code for estimating bone marrow adiposity (BMA) from a 3T head MRI scan

Overview of scripts and their inputs and outputs

The main files for using the trained model:

- **mainscript.sh** for extracting intensities (head surface, find calvarium, vertex intensities)
  - Input: nu.mgz files from a given freesurfer folder
  - Output: intensity matrix
- **run_predBM.py**
  - inputs: **Model** + intensity matrix
  - output: localised layer boundaries
- **computeAuxiliaryFiles.r:**
  - inputs: intensity matrix + boundary matrix
  - outputs:
    - layers-by-datapoint intensity plot (with boundaries)
    - summary statistics at datapoint level for intensity of outer bone, bone marrow, inner bone
    - QC plot for visualisation of outlier datapoints
    - Intensity statistics across datapoints

Additional files for simulating new training data and training a new model:

- **bmSimulation_50_versionA.r**
  - Inputs: parameters set in script
  - Outputs:
    - Intensities
    - Boundaries between anatomical layers
- **bmDetectNN50_randomAux_training_A_reduced.ipynb** (colab notebook)
  - Inputs: simulated intensities and ground truth boundaries (from previous script)
  - Output: trained NN model (which can be used by predBM.py)

Details of scripts and their inputs and outputs

Important: this code is designed to work on T1w head MRI scans following intensity normalisation. This is usually done via freesurfer standard processing (see: https://surfer.nmr.mgh.harvard.edu/fswiki/recon-all), e.g:

```
recon-all -s $SUBJECT -i $T1 -sd $SUBJECTS_DIR -all
```

The procedure for locating the BM and measuring its T1-weighted signal intensity involves the following steps:
1. identifying the skin surface of the head
2. identifying the calvarial part of the head using the top of the cerebellum and the accumbens as reference points
3. for each point of the calvarial surface extracting an intensity array 25 mm into the skull (each layer is 0.5 mm thick, so 50 layers).
4. applying an artificial neural network to identify which layers of the intensity array correspond to BM (Figure 1). This network has 50 input nodes (one intensity for each layer) and 6 output nodes (upper and lower bounds for each of outer table, BM, inner table).
5. Using the network-estimated location of the BM within the intensity array of a datapoint, we averaged these intensities to obtain the intensity for the datapoint, and then averaged these across all datapoints of the calvarium to produce the global BMA measure for the scan.

The scripts that implement this procedure are dependent on the following software environment:

- Matlab: 2018a
- Freesurfer: 5.3.0
- R: 3.5.0 + packages: ggplot2, pals
- Python: python 3.8.1 + packages: tensorflow 2.4.1, numpy 1.19.5

Steps 1, 2, and 3 - Extracting intensity arrays

Overall purpose:

1. Identify skin surface using watershed https://surfer.nmr.mgh.harvard.edu/fswiki/mri_watershed.
2. Identify calvarium based on brain topology: compute mask in Matlab by reading in the parcellation from freesurfer ("aseg" file) and identifying the relevant region borders for masking the image.
3. Extract intensity array for every calvarium datapoint: intensity values are extracted from nu.mgz in a stepwise run from the outer skin surface inwards using mri_vol2surf from Freesurfer

Scripts: mainscript.sh, compileLayers.m and makeIndividualMask.m

- Place all scripts in the same directory.
- The two matlab scripts do not need to be edited.
- Edit mainscript.sh to set up the specifics of your software environment (fsl, freesurfer, matlab), adjust the subject name and change the paths to the folders (data, results, scripts).

Inputs:

- The script works on freesurfer processed subject level folders. Once you run the mainscript, it takes the nu.mgz files from a given freesurfer folder as input

Outputs:

- Outputs a text file containing an intensity matrix. This matrix has 50 rows (one per layer) and nbDatapoints columns.

Calling scripts:

```
sh mainscript.sh
```

Nota Bene:

This intensity extraction procedure may fail to produce an output for some scans or produce a scan with less than 5000 datapoints (which is an indication of a poor quality scan), in which case it is not possible to continue with the analysis.
If the intensity extraction procedure completes successfully, the rest of the code should run successfully. However, users will need to perform basic quality control checks as described in the paper.

Unfortunately, we do not have the capacity to answer questions regarding:

- how to install the environment required by the scripts: Matlab, FreeSurfer, R, Python. Hopefully, your local IT support can help.
- why specific scans may fail to produce output.

Step 4 - Localise boundaries of bone marrow (and inner and outer bone)

We use the run_predBM.py script to predict (for each datapoint) the upper and lower bounds for each of the outer bone, bone marrow, and inner bone (6 bounds in total).

Inputs:

- Directory containing the neural network model (this directory should contain the model in a file named "saved_model.pb")
- CSV file containing the intensity matrix (see above)

Output:

- A text file containing the predicted bounds matrix (with 6 rows and nbDatapoints columns). The output file will be written to the same directory as the input intensity matrix and will have the same file name with the suffix "_ObBmIb_bounds.txt"

Calling the script:

```
python run_predBD.py —model="NNmodels/2021-02-11" pathToIntensityMatrix.csv
```

Step 5 - Compute summary statistics (at datapoint and calvarium levels) and generate plots

We use the computeAuxiliaryFiles.r to compute to compute statistics and plots:
Inputs:

- CSV file containing the intensity matrix (see above)
- CSV file containing the bounds matrix (see above)

Outputs:

- A plot of the entire “layers by datapoints” intensity matrix together with the location of the predicted bounds (file suffix “_structData_calvarialVertices_intensitiesAndBounds.png”)
- Summary statistics at the datapoint level (file suffix "_structData_calvarialVertices.csv”). There are 4 structures: ob (outerbone), bm (bone marrow), ob (outerbone), all (all layers). And, for each of these structures, there are 6 statistics of interest: IntMean (intensity mean), intSum (intensity sum), intSd (intensity standard deviation), thick (number of layers thick), first (first layer), last (last layer). This gives a total of 24 columns (4 x 6). There is a 25th column (MD) containing the Mahalanobis Distance of each datapoint in the bmFirst vs. BmIntMean space (see article for details).
- A plot of all datapoints in the bmFirst vs. BmIntMean space for visualising the MD outliers (file suffix "_structData_calvarialVertices_bmFirstVsBmIntMean.png")
- Statistics (nbDatapoints, min, median, mean, max, sum, sd) across datapoints for each of the 25 statistics (produced at the datapoint level): (“_structData_calvarialVertices_summaryStats.csv")
    - The most important statistic is the mean of bmIntMean which is our measure of calvarial BMA for the scan.
    - Other statistics were used in quality control to determine whether a scan had been correctly processed. Briefly, for a scan to be considered to have passed QC, it had to: have more than 5000 calvarial datapoints AND have standard deviation of the intensity of the outer table below 30 (sd of obIntMean < 30), no more than 0.5% of a scan’s datapoints had an MD > 25. See the paper for further details. The user will have to perform quality control themselves (using the data contained within the files described above) and may wish to use different QC criteria.

Calling the script:

```
Rscript computeAuxiliaryFiles.r pathToIntensityMatrix.csv pathToBoundsMatrix.csv
```

Step 0 - Training of the neural network

As long as this code is being applied to MRI data with the required characteristics (see above), it should not be necessary to train a new neural network model. However, for documentation purposes, we provide the data simulation and model training code.

We trained the network on simulated data incorporating the natural intra- and inter-individual variation in thickness and intensity of the relevant anatomical structures (Figure 1A).

Code for simulating intensity arrays: bmSimulation_50_versionA.r
Input:

- parameters are hard-coded at beginning of file (you will need to set the output directory)

Outputs (file names prefixed by the input parameters: mean bone thickness, mean bone marrow intensity, mean body fat thickness):

- file with 50 intensities for each datapoint (file suffix "_intensitiesNorm.tab”).
- file with 6 boundaries for each datapoint (file suffix "_boundsTrueNorm.tab"): upper and lower bounds of each of the anatomical structures of interest (outer bone, bone marrow, and inner bone).

Code for training a new model: bmDetectNN50_training_A_reduced.ipynb
(https://colab.research.google.com/drive/16ecbr3L3qHcX2bjg2SuRY2gdCqK8Rf0k)
Input:

- simulated intensity data and corresponding ground truth (see above)

Output:

- Trained model: saved_model.pb

We recommend running this script on google colab. In order to do so, you will need to:

- Make the files output from the simulation code available on google colab
- Modify the file paths in the script to point to these files and to the directory where the output model should be saved.