Instructions for Running fNIRS data through Image Recon Pipeline using NeuroDOT

17 December 2019

# Rotation and segmentation

## Rotation

* We use the *sub\_neurologicalHighRes.nii* files for each subject
* This first needs to be rotated so that it lands on close to an axial orientation for NIH – not true for Gates - Vince’s script does the final rotation after that.
* *MRI\_Rotate.sh* does this first rotation, but you need to change three things:
  1. The subject list
  2. The working directory
  3. The -rotate x y z amount. It can be negative to allow rotation the other direction.
* This will need to be done on an individual basis for all participants, then we can move to the next step.

## Segmentation

* We need to do this in two steps (for NIH) - one is to remove the background noise, the other is to strip the skull
* These can be run as a group
* Turn to the *MRI\_cleanup\_May18.sh* (in the same folder as *autoSegment20190428.sh*) – if running India Gates data the script just has step 6 in it, ignore the in-between ones.
  1. Subjects need to be put in the subject list
  2. Change the working directory
  3. First argument changes the step amount. We probably want step(a-350), but this can be adjusted if needed.
  4. The next two arguments shouldn’t need adjustment - they remove the outside noise, and create ${subj}\_final\_Masked.nii.
  5. We then multiply the image by itself. Check this with Sam - it should be necessary for all NIH kids where resolution is poorer, but this is not necessary for Gates phase 2.
  6. The last line then needs adjustment - shown below. Make sure the wd is correct, then there are two changes you can make: One is the 0.7 below can be adjusted up or down - this is the segmentation parameter. The other is you can add a -a at the end - this uses the other segmentation method (useful if the first one is leaving skull in).

autoSegment20190428.sh -t /Users/administrator/Desktop/test/${subj}\_enhanced.nii -o  
//Users/administrator/Desktop/test/${subj}/ -s 0.7 -c 3 -b -m

autoSegment20190428.sh -t /Users/administrator/Desktop/test/${subj}\_enhanced.nii -o  
//Users/administrator/Desktop/test/${subj}/ -s 0.6 -c 3 -b -m -a

* The end result of this should have no skull, and needs to be checked that it has not cut away part of the brain. Equally there needs to be padded skull all around the brain (although Vince’s script should take care of that). Check previous examples so you know what this means.

1. Running MCs in AtlasViewer

# Photon migration simulation preparation

## Individual anatomy

* Once segmented you need to open MATLAB, and navigate to the folder with all the segmented images for that one participant in. You do not need a NIRS or a digpts at this stage.
* Open *AtlasViewerGUI* (by typing the same into MATLAB)
* It will popup with a dialogue box - just hit close or cancel.
* When it opens AV, hit file -> import MRI anatomy
* By default, it should populate the fields - make sure hseg is with segmented head, WM is with white matter, brain is with grey matter, and CSF is with CSF.
* Hit import
* Once this is done, it will ask you to select reference points - hit yes
* These should be the same as for a digitization - NZ, IZ, CZ, LP, RP
* You *MUST* check during this step that the head is aligned with the cardinal left right as given. If not, the rotation and segmentation was not right.
* When this is done it will ask if you want the 10-20 EEG points, hit yes
* When this is done, you can close AV

## Move files

* If you already have an atlas or anatomical (ie template), start here.
* Make a new subfolder for the participant entitled *digitization*
* Copy the anatomical folder, the digpts, the NIRS file, and the HeliumScripts folder into there.
* Use the digpts from that session - if it doesn’t exist, use the template with the same capsize
* NIRS does not need to be from that subject, just the right number of sources/detectors. You need to check if this is right regarding short sources as well.

## Set up probes

* In MATLAB navigate to the digitization folder and open AV
* Go to tools -> register atlas to Digpoints
* Then at the bottom Register Probe to Surface
* Go to Forward model -> set MC Parameters
* Check scattering values with John before going ahead with defaults. These are found in the optical properties excel sheet and can be copied across for NIH and Gates. Change these to what’s specified in the right hand columns.
* Number of photons: 100000000 (thats *8* zeros - count them!)
* Forward model -> Generate MC Input
* Then cancel (don’t run locally on your computer)
* This gives fw …..inp files within the fw folder - check you have the right number
* IF you are running locally to test, you can say ‘file exists, try running now,’ and wait for it to finish. Once that is done you can move to complete steps, below.

# Running MCs

## Transfer files to HPC

* This is only relevant if running on the HPC. Make sure Homer2 versions on HPC match what you have on the computer.
* Copying folder from local destination to HPC, using your ID:

scp -r /Users/administrator/Desktop/sub1002  
taw15kfu@hpc.uea.ac.uk:/gpfs/home/taw15kfu/

* I would then advise using the linux to see them directly (open the HPC app, and hit files). If you prefer to do it remotely, you sign in with:

ssh -XY (id)

* After copying homer2 over, (if not there already - make sure version is the same as the setup computer), set permissions, this allows homer to access the tMCimg file for monte carlo:

chmod 700 /gpfs/home/taw15kfu/homer2/PACKAGES/tMCimg/bin/Linux/tMCimg

* You may also need to make the tMCimg first for the HPC

## Changing paths in files

* We have the fw folder and the HeliumScripts folder (again for HPC use).
* Make changes to the pathnames with the two files, and the subject lists (can do multiple).
* Every single inp file the path needs to be changed
* HeliumScripts folder will be there with all the bsub files
* Also in HeliumScripts, the bsub files need paths to the Headvol to be updated
* *change\_inp.sh* does this, use the backslash vs forward slashes to separate out words, and the second part tells you what each one gets replaced with
* *change\_bsub.sh* does the same thing for the bsub files
* Check that everything is right before proceeding
* If working remotely - use nano to see it in editor instead of bash
* Run these two scripts and check the output.

bash change\_bsub.sh

bash change\_inp.sh

## Running MC

* To run, first edit the subject list in *sub\_job.sh*

bash sub\_job.sh

* Check bjobs, if it has run suspiciously quickly then there is a problem
* Once it’s done, bjobs will show no active jobs
* Then we can transfer everything back to the local destination

## Move back and complete steps

* Once everything finishes, copy folder from HPC to local destination (reverse of the above):

scp -r taw15kfu@hpc.uea.ac.uk:/gpfs/home/taw15kfu/NIRS\_Project/  
/Users/administrator/Desktop

* Do the register atlas to digpts and register probe to surface again.
* After that, you need to cd to the *digitization/fw/* folder of that participant in the terminal and run the following (unless run locally):

touch \*

* This updates the files so they are modified recently. Without this it might say input is newer than output, and wipe the output, or ask you to rerun it.
* If you try to run with Forward model -> Generate MC Input, it will say there is already an input, so hit OK
* *IMPORTANT* if it says anything else hit cancel and check that the fw folder contains the new MC files
* Tick Forward model -> enable sensitivity matrix volume
* Then go to Forward model -> Generate load sensitivity profile
* These will take a while to load. It may ask if the head needs downsampling, if it asks, say yes, and accept the downsampled head.
* This generates a coloured picture which is your sensitivity profile
* Before closing, check that the *Adot* files are in the *digitization/fw* folder

1. **Create Input File for NeuroDOT pipeline**

* There is a template in the ‘scripts’ folder called ‘SubjectList\_Template.xlsx’. It is best to start creating the input file in Excel to allow for easy updating.
  + The Excel template uses data in the subjectID column to fill in the correct path details. It also uses the MC data in column C to fill in the correct path for the headvol file in column F. **So be sure to keep the formulas intact.**
  + To edit the file, paste in your subject list in column A and delete any extra rows.
  + Next, update the paths in each column in row 1 and ‘fill down’. Note that data might be in different places, so it is easy to just ‘fill down’ the relevant rows with different paths as needed. For instance, if some kids use templates, you can group them together and then point to a single template (headvol) file in column F.
* Once the file has been updated, save it as a .csv file. Then open the .csv with a text editor like BBEdit or TextWrangler. Find the commas and replace with space. Then save as a .prn file. **Be sure to specify that line breaks are set in Unix format.**
* Column specification:
  + SubjectID
  + Full path to template .nirs file for the project (e.g., India.nirs, HWB.nirs)
  + Full path to subject-specific MC folder
  + Full path to the NIRS folder (created by NIRS\_Pro1)
  + Full path to the desired output folder (where you want the ImageRecon files to be written)
  + Full path to the headvol file for this subject (e.g., headvol.vox – could be subject-specific or a template)

1. **Creating sensitivity profiles in niftii format for NeuroDOT Light Model**

* Run **transformSensProfileToAnat\_NeuroDOT.sh**.
* This script takes a subject list file (see input file above).
* Comments:
  + This script differs from the original in that it does not trim the sensitivity data to the brain. Rather, it uses the entire profile including the scalp and surface tissues.
  + The script also differs in that it packs all of the sensitivity data into a single .nii file with sub-bricks by channel (in an order corresponding to the measurement list in the .nirs file).
  + A light thresholding is applied to the sensitivity volumes (0.000001). This was useful because a few volumes fill the entire brain if left un-thresholded.
  + We also downsample to 2mm resolution to keep the file size reasonable.
* To run the script, do the following:
  + Navigate to the ‘Data’ folder for the project in terminal. This is where your subject list file should be located.
  + Link in the Github scripts: example…

export PATH=${PATH}:/Users/nfb15zpu/Documents/GitHub/MRI-NIRS\_Pipeline/scripts

* + Link in matlab: example…

export PATH=${PATH}:/Applications/MATLAB\_R2018b.app/bin

* + Run: transformSensProfileToAnat\_NeuroDOT.sh {filename} – example…

transformSensProfileToAnat\_NeuroDOT.sh SubjectList30mo\_NIHVWM\_ICPipe.prn

1. **NIRS Processing in EasyNIRS / Homer2:**

* First step in NIRS processing is to sort the stim marks. This is done in some variant of NIRS\_Pro1.m
* To make sure there are no Infinity values in the .nirs file, the following scripts can be run. Note that this was observed with the NIRS machine in India, but not at UEA.
* FindVal.m : it creates a text file with first and last time points with Infinity values.
* Examine these text files. For those .nirs files that have Infinity values only at the end – I remove the cells with ReplaceVal.m
* What is left are files with Infinity values in the middle of the recording. Two things needs to be done to these .nirs files. First, the Infinity values have to be replaced with interpolated values. For this, first run FinalValues.m to create text files that show the timepoints for all the Infinity values. You need to use this to change values (check comments to know whether you need to replace 25 points or 50 points or more) in Inter\_val.m for each file manually.
* After interpolation, the stim triggers within 10 timepoints of these segments need to be turned off. Run Turnstim.m to do this.
* After correcting the NIRS files, they need to be analysed using EasyNIRS. This is done using a configuration file in EasyNIRS (.cfg file) that loads the specific processing options desired. Here’s a typical sequence:
  + hmrIntensity2OD [converts to optical density]
  + enPruneChannels SD [prunes channels with low signal-to-noise]
  + hmrMotionCorrectPCArecurse [motion correction using targeted PCA]
  + hmrMotionArtifactByChannel [flags epochs that have uncorrected motion artefact]
  + enStimRejection [turns ‘off’ stim marks where HRF window falls within a motion epoch]
  + hmrBandpassFilt [band pass filter]
  + hmrOD2Conc [converts optical density to concentration data] – this last step is useful for viewing data in EasyNIRS but it technically not needed by NeuroDOT (which only uses the procResult.dod data)
  + Note: we also often run a GLM in EasyNIRS using hmrDeconvHRF\_DriftSS. This returns beta values for each channel. Again, NeuroDOT doesn’t use this info, but it can be useful for sanity checking the GLM run in NeuroDOT.

1. **Image Reconstruction using NeuroDOT**

* Open matlab
* Navigate to Data folder
* Make sure the Github ‘scripts’ folder and subfolders are linked into the matlab path (check the ‘set path’ toolbar).
* Make sure the Github ‘NeuroDOT\_Beta’ folder and subfolders are linked into the matlab path.
* Run **ImageRecon\_NeuroDOT.m** in matlab. Example…

ImageRecon\_NeuroDOT('SubjectList30mo\_NIHVWM\_ICPipe.prn', 25)

* Comments:
  + The ‘25’ specifies the sampling frequency desired in the final 4D image recon file. If matlab crashes because it runs out of memory, you’ll have to downsample the data from, say, 25Hz to 10Hz.
  + To make the image recon data as trim as possible, we only reconstruct data from 20s before the first stim to 20s after the last stim.
  + The output of this step will be a 4D image reconstructed file for each run with HbO (voxels x time), HbR (voxels x time), and the metadata. This is in a matlab data structure to save space.

1. **Running GLMs using NeuroDOT**

* Open matlab
* Navigate to Data folder
* Make sure the Github ‘scripts’ folder and subfolders are linked into the matlab path (check the ‘set path’ toolbar).
* Make sure the Github ‘NeuroDOT\_Beta’ folder and subfolders are linked into the matlab path.
* Run **RunGLM\_NeuroDOT.m** in matlab. Example…

RunGLM\_NeuroDOT('SubjectList30mo\_NIHVWM\_ICPipe.prn', [1,2,4,5], 250, ‘WLRN’, 25)

* Comments:
  + Input 1: subject list input file
  + Input 2: regressor list you want included in this particular GLM
    - Regressors are specified in the ‘s’ matrix in NIRS\_Pro1. We typically code up many more events than is included in any particular GLM. The regressor list specifies which ones to include.
  + Input 3: the duration of each event (the duration of the boxcar in the GLM). This is specified in time steps (so duration \* sampling frequency). As an example, the duration of a trial in the VWM task is 10 s, so we would enter 250 (10\*25Hz). In HWB, each naming event is short, so we would enter 25 (1s\*25Hz). Note: if the data has been downsampled, the duration should be in the downsampled frequency.
  + Input 4: a tag for each output filename to indicate the ‘name’ of the GLM you are running (e.g., ‘WLRN’ for ‘word learning GLM’).
  + Input 5: the sampling frequency used during image recon (e.g., 25Hz if you didn’t have to downsample; or 10Hz if you did)
* This script will output betamaps for each condition for oxy and deoxy in the headvol space (i.e., subject space).

1. **Move to Group Space (e.g., CustomMNI)**

* Navigate to Data folder in terminal
* Link in the Github scripts: example…

export PATH=${PATH}:/Users/nfb15zpu/Documents/GitHub/MRI-NIRS\_Pipeline/scripts

* Link ANTS tools: example…

export PATH=${PATH}:/Applications/ANTS/bin

* Run registerCommonDriver.sh. This takes the input file as input:

registerCommonDriver.sh [inputfile name]

* Note that lines 71 and 72 need to point to the Subject-specific T1 and Brain mask you want to use for the registration.

1. **Create a Group Mask**

* Github ‘scripts’ has an example called ‘NIRS\_Mask\_ANOVA\_Example.sh’ that can be edited. The goal of this script is to sum up how many subjects contribute at least some data (across any condition) to each voxel. This can then be thresholded in the final step based on a percentage of subjects (e.g., 60-70%).
* Once the mask has been created, it can then be multiplied with all the group-level images to filter out any ‘fringy’ voxels with sparse data.

1. **Group ANOVA**

* Sample script: Load\_MVM.txt.
* Include -GES option to get effect size estimates
* Include -resid option to estimate the spatial autocorrelation (ACF)
* If one or more within-subjects variables, include -wsMVT to get multi-variate estimates of the within-subjects effects
* Include -SS\_type 2 (a bit more powerful)
* Example:

3dMVM -GES -prefix GatesPLAgeSSHb\_gesConHb3 -jobs 2 \

-resid GatesPLResidConHb3.nii \

-mask Infants\_IntersecMask.nii \

-bsVars 'Age' \

-wsVars "SS\*Hb" \

-wsMVT \

-SS\_type 2 \

-dataTable \

1. **Concatenate ANOVA results**

* Sample script: conCat\_ANOVA\_Example.sh
* This prepares the data for the next steps
* Basically, packs the beta images into a single file for each beta map you are contrasting in the ANOVA and each chromophore.

1. **Determine thresholds for family-wise error correction**

* Now we need to correct for multiple comparisons. This involves two steps.
* First, you need to estimate the spatial noise in the data. Do this by running an analysis on the residuals from the ANOVA.

3dFWHMx -input GatesPLResid.nii -mask Infants\_IntersecMask.nii -acf

* This computes the spatial auto-correlation in the data (residuals) and fits parameters to estimate the ACF (see Cox et al., 2017). The result is a nice plot of the ACF and 3 output parameters estimating the ACF.
* Next, use the ACF parameters to run 3dClustSim. This will set a cluster size threshold. Only clusters >= the size threshold can be considered significant after controlling for family-wise error.
* 3dClustSim requires setting a voxelwise p value and an alpha level. Based on Cox et al., 2017, we’ll use p = 0.01 and alpha = 0.05. Example:

3dClustSim -mask Infants\_IntersecMask.nii -acf 0.736355 6.45424 2.94415 -pthr 0.01 -athr 0.05 -iter 10000 -nodec -quiet

* This command will run simulations estimating the cluster size threshold for different types of ways to define the cluster. See the AFNI help for details. As an example, we might select NN1 (first nearest neighbor clustering = voxels cluster if faces touching), two-sided (non-directional hypotheses), and not bi-sided

1. **Threshold the ANOVA results**

* Run 3dClusterize (replaces old 3dclust and 3dmerge).
* Sample script: 3dClusterize\_Child\_WLCT.sh
* This applies criteria from 3dClustSim. The example code shows how to loop through all the effects of interest.
* 3dclusterize takes the input image, specifies the sub-brick to threshold (see ithr), takes the mask input, NN desired, number of voxels specified by 3dClustSim (clust\_nvox), and saves the data as clusters with increasing numbers (pref\_map).
* -1sided RIGHT\_Tail : since we are using F stats, this should be the F critical determined by using the cluster threshold in the AFNI gui. It is 1sided here because F is always positive, but the critical F is conceptually two-sided. Ask john for details.
* The code also writes the GES stats (when relevant)

3dROIstats -mask Infants\_cPL\_gesConHb3/0.01/${effect}/clust\_order\_${effect}\_l1\_01.nii GatesPLAgeSSHb\_gesConHb3+tlrc'[16]' > \

Infants\_cPL\_gesConHb3/0.01/${effect}/${effect}\_l1\_01\_ges.1D

* Additional helpful commands. Whereami gives labels from the MNI atlas if you have group data in that space. This reads in the values in columns 13, 14, 15 from the 1D file created above (note: indices start at 0 in AFNI, so columns 14, 15, 16).

whereami -coord\_file Infants\_cPL\_gesConHb3/0.01/${effect}/${effect}\_l1\_01.1D'[13,14,15]' -tab -space MNI > Infants\_cPL\_gesConHb3/0.01/${effect}/${effect}\_l1\_01\_wai.1D

* Finally, it’s useful to copy the underlay image to your directory so you can view the clusterized .nii file.

cp MNI\_avg152T1+tlrc.BRIK Infants\_cPL\_gesConHb3/0.01/${effect}/

cp MNI\_avg152T1+tlrc.HEAD Infants\_cPL\_gesConHb3/0.01/${effect}/

1. **ROIStats on ANOVA results**

* Sample script: ROI\_stats\_Example2.sh
* This computes the average beta values for each cluster for each participant. These data can then be used to run, for instance, correlations with behavioural variables.
* The example code loops through all participants (in each age group or overall, depending on how you’ve set up the data). It also loops through all the effects included in the previous step.
* The key call is to 3dROIstats. A key parameter here is the list of bricks that need analysing. The brick list is [0,1,2,3,4,5] below. This list was created when we concatenated the file in step 10. So in the example below, I’m averaging the beta values for 0 = oxy load 1, 1 = deoxy load 1, 2 = oxy load 2, 3 = deoxy load 2, 4 = oxy load 3, 5 = deoxy load 3.

3dROIstats -mask Infants\_cPL\_gesConHb3/0.01/${region}/clust\_order\_${region}\_l1\_01.nii Infants\_cPL\_gesConHb3/Concat\_4mo\_${names}.nii'[0,1,2,3,4,5]' > \

Infants\_cPL\_gesConHb3/0.01/${region}/ROIstats/${region}\_4mo\_${names}.1D

1. **CombinedROIStats**

* Sample script: CombinedROIStats\_Example.m
* This organizes the output of ROIstats into a file with the average beta values for each cluster for each effect for each participant. These data can then be used to run, for instance, correlations with behavioural variables.
* The example code loops through all participants and all effects. The output is a long form .csv file that can be read into R for follow-up analyses.
* Note: if you have effects with no clusters, it is useful to change the effect folder name, adding ‘\_NoClusters’. Otherwise, this code with open the files created and throw an error because there are no data in the subject-specific 1D files.

1. **Extract Time Series Data from Image Recon Files from significant clusters**

After group analysis, it can be useful to extract individual time series from the significant clusters to show a block average time series that NIRS folks are used to. Here’s how to do this.

1. Start with the clust\_order\*.nii file for the effect you want to examine

* e.g., clust\_order\_6Hb\_l1\_01.nii
* copy these ‘clust’ files to a new folder that will store the output of the current analysis

1. Create a new .prn file, adding a new column to the original .prn file that specifies the full path of the output folder (the new folder you created in step 1)
2. Run registerCommonDriver\_Invert.sh (or registerCommonDriver\_Invert\_wFixOrientation.sh if you are working with a data set that has issues with the orientation…Sara’s data has this; I think Courtney’s data as well).

* This resamples the headvol for each subject to 2mm resolution
* Then it creates a subject-specific clust file by inverting the original affine transformation applied to the data to get to group space. So the end result is a mask with the clusters in each ‘clust’ file moved to subject-specific space.
* If you are using the ‘fixOrientation’ version, a second set of ‘prefix’ files will be generated. These are temp files (that can ultimately be deleted).
* Example:

bash registerCommonDriver\_Invert.sh Y1\_finalComboSubjListGroup.prn

1. run ExtractHbFromMask.m

* To be used for extracting a weighted average time series for each subject for each effect from each cluster for each regressor for each chromophore
* Input example in matlab

ExtractHbFromMask('Combo\_finalSubjListGroup.prn',[1,2,3],'Test',10,10,3,0,0)

* Input file name (.prn file)
* Regressor List – the conditions from the original GLM that you want to extract block averages for
* A name for the analysis (for the output file). The name is also a column in the .csv file which can be useful for tagging ‘Y1’ vs. ‘Y2’ (e.g., if you ultimately want to combine across years)
* newSamplingFreq: the sampling frequency of the image recon files
* HRFDuration: the duration of the HRF window (in seconds)
* MaxMaskValue = the max mask cluster value across all clust\_order files to analyse
* checkAlignment – flag to display the headvol, the cluster mask, and the ‘GoodVox’ as overlays. Useful for checking a handful of subjects initially, but set to 0 for full runs
* showHRF – flag to plot the resultant time series data for each subject. Again, most useful for initial explorations, but set to 0 for full runs.