Jan 12, 2026

Quantifying Amyloid Burden Across the Alzheimer's Continuum: A Reproducible Centiloid Pipeline For Analysis in Resource-Constrained Settings

In 1 collection

DOI
https://dx.doi.org/10.17504/protocols.io.81wgbwzjqgpk/v1

Abdullahi Adeniyi[1], David Oladeji[2], Chinyere Martina Anene-Ogbe[3], Babari a Babari, Michelle Freddia[4], Chijioke Kennedy Ibe[5], Nnennaya Chinaemerem Caroline Nwasike[6], Nkenganyi Aka Elvira[7], Alfonso Fajardo v[8], Philip Nkwam[9], Oluwatobi Iyanuoluwa Akinmuleya[10], Udunna Anazodo[11], Abdalla Z Mohamed[12], Ethan Draper[13], Martina Anene-Ogbe[14]

[1]College of Health Sciences, Obafemi Awolowo University, Ile-Ife.; [2]College of Medicine, University of Lagos.; [3]Department of Human Anatomy, University of Port-Harcourt, Nigeria; [4]University of Yaounde 1, Cameroon; [5]Department of Medical Radiography & Radiological Sciences, University of Nigeria Nsukka, Nigeria.; [6]Faculty of Medicine and Biomedical Sciences, University of Yaounde 1, Yaounde, Cameroon & Brain Research Africa INitiative, BRAIN, Yaounde, Cameroon; [7]University of Buea, Cameroon; [8]Montreal Neurological Institute, McGill University, Montreal, Canada; [9]College of Medicine, University of Lagos; [10]Shenyang Medical College, Shenyang, China; [11]Montreal Neurological Institute, McGill University, Montreal, Canada.; [12]United Arab Emirates University, UAE; [13]Department of Bioengineering, Imperial College London; [14]University of Port Harcourt

Capstone_Project_PET_A | CONNExIN (COmprehen...

Martina C Anene-Ogbe

OPEN ACCESS

DOI: https://dx.doi.org/10.17504/protocols.io.81wgbwzjqgpk/v1

Protocol Citation: Abdullahi Adeniyi, David Oladeji, Chinyere Martina Anene-Ogbe, Babari a Babari, Michelle Freddia, Chijioke Kennedy Ibe, Nnennaya Chinaemerem Caroline Nwasike, Nkenganyi Aka Elvira, Alfonso Fajardo v, Philip Nkwam, Oluwatobi Iyanuoluwa Akinmuleya, Udunna Anazodo, Abdalla Z Mohamed, Ethan Draper, Martina Anene-Ogbe 2026. Quantifying Amyloid Burden Across the Alzheimer's Continuum: A Reproducible Centiloid Pipeline For Analysis in Resource-Constrained Settings. **protocols.io** https://dx.doi.org/10.17504/protocols.io.81wgbwzjqgpk/v1

Protocol status: Working
We use this protocol and it's working

Created: August 29, 2025

Last Modified: January 12, 2026

Protocol Integer ID: 225757

Keywords: Dementia, Neuroimaging, PET, Brain Imaging Data Structure, PET-BIDS, PET Quality Control, PET Neuroimaging, BIDS, Positron Emission Tomography, quantifying amyloid pet, quantifying amyloid burden, amyloid pet, gaain centiloid scale, gaain result, published gaain result, gaain full dynamic dataset, standardised uptake value ratio, t1 mri data, weighted mri, reproducible centiloid pipeline for analysis, alzheimer, gaain, reproducible centiloid pipeline, hybrid fsl

Abstract

Overview

This protocol describes two reproducible cloud-based pipelines (FSL-based and hybrid FSL/SPM-based) for quantifying amyloid PET using the GAAIN Centiloid scale. The pipeline processes PiB PET and T1 MRI data to calculate standardised uptake value ratios (SUVR) and Centiloid values.
These pipelines produce results of very robust correlation with the values from the Centiloid Project.

Key Features

- Open-source FSL implementation (no SPM required)
- Validated against published GAAIN results
- Quality control steps included
- Batch processing capable

Software Requirements

Required software
FSL 6.0+ (https://fsl.fmrib.ox.ac.uk/fsl/fslwiki)
SPM12
Neurodesk environment (recommended) or Linux/MacOS
bash shell
Basic command-line familiarity

Data Requirements

GAAIN Full Dynamic Dataset (available at https://gaain.org)
AD patients: AD01-AD25
Young controls: YC101-YC134
GAAIN 50-70 Minute Dataset (NIFTI format)
T1-weighted MRI (NIFTI format)
VOI masks: voi_ctx_2mm.nii, voi_CerebGry_2mm.nii

Attachments

Framing Info File: framing_info.csv 1.1KB
Hybrid FSL/SPM scripts:

- Data structure- 01_data_org.sh 3.9KB
- Preprocessing- 03_preprocessing.sh 9.3KB
- Visual QC- visual_qc.sh 2.2KB
- Analysis- analysis.sh 2.9KB
- Statistical Analysis- statistical_test.py 5.5KB

- HTML aggregate for QC- generate_report.py 3.7KB

Troubleshooting

Problem
Cerebellar values too high (>10);VOI misalignment
Solution
"Check affine registration, visualize alignment"

Problem
PET values extremely high (>100);Intensity scaling issue
Solution
"Check original PET units, may need scaling"

Problem
Registration fails;Poor image quality
Solution
"Check BET extraction, adjust -f parameter"

Problem
SUVR ~1.0 for AD subjects;VOI misalignment
Solution
Ensure proper VOI alignment to PET space

Problem
Memory errors;Large images
Solution
Use fslchfiletype to convert to NIFTI_GZ

Problem
Coregistration failure
Solution
Check image orientations with fslorient -getorient

Problem
Low correlation values
Solution

Review QC images for specific subjects

Problem

Missing frames

Solution

Verify frame range in framing_info.csv

Problem

Disk space errors

Solution

Enable cleanup in the preprocessing script

FSL PIPELINE

1 Environment Setup

1.1 Step 1: Launch Neurodesk with FSL or ensure FSL is loaded

```
module load fsl  # or source ${FSLDIR}/etc/fslconf/fsl.sh
```

1.2 Step 2: Set MNI template path (adjust for your system)

```
export
MNI_TEMPLATE="/cvmfs/neurodesk.ardc.edu.au/containers/mrtrix3_3.0.
1_20200908/mrtrix3_3.0.1_20200908.simg/opt/fsl-
6.0.3/data/standard/MNI152_T1_2mm.nii.gz"
```

1.3 Step 3: Verify FSL installation

```
which flirt
which bet
which fslmaths
```

2 Single Subject Processing Pipeline

2.1 Step 1: T1 MRI Processing

- Define subject ID

```
SUBJECT="AD01"
```

- Skull stripping

```
bet data/${SUBJECT}/anat/${SUBJECT}_MR.nii \
    data/${SUBJECT}/anat/${SUBJECT}_MR_brain.nii.gz \
    -f 0.3 -B -R
```

- T1 to MNI normalization (12 degrees of freedom)

```
flirt -in data/${SUBJECT}/anat/${SUBJECT}_MR_brain.nii.gz \
      -ref ${MNI_TEMPLATE} \
      -out data/${SUBJECT}/anat/${SUBJECT}_MR_MNI.nii.gz \
      -omat data/${SUBJECT}/transform/T1_to_MNI.mat \
      -dof 12
```

2.2 **Step 2: PET Processing**

- PET to T1 coregistration (6 DOF, mutual information)

```
flirt -in data/${SUBJECT}/pet/${SUBJECT}_PiB_5070.nii \
      -ref data/${SUBJECT}/anat/${SUBJECT}_MR_brain.nii.gz \
      -out data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_T1.nii.gz \
      -omat data/${SUBJECT}/transform/PET_to_T1.mat \
      -dof 6 -cost mutualinfo
```

- PET to MNI normalization (concatenate transformations)

```
convert_xfm -omat data/${SUBJECT}/transform/PET_to_MNI.mat \
            -concat data/${SUBJECT}/transform/T1_to_MNI.mat \
                    data/${SUBJECT}/transform/PET_to_T1.mat

flirt -in data/${SUBJECT}/pet/${SUBJECT}_PiB_5070.nii \
      -ref ${MNI_TEMPLATE} \
      -out data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_MNI.nii.gz \
      -applyxfm -init data/${SUBJECT}/transform/PET_to_MNI.mat
```

- Intensity thresholding (0.001 as per GAAIN)

```
fslmaths data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_MNI.nii.gz \
         -thr 0.001
data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_MNI_thr.nii.gz
```

3 Intensity Normalization

3.1

- **Step 1: Check for Intensity Issues**

```
# Check cerebellar values before processing
REF_CEREBELLAR=3.5  # Target based on validated subjects


for subject in AD* YC*; do
    if [ -f "data/$subject/pet/${subject}_PiB_5070_MNI.nii.gz" ];
then
        cerebellar=$(fslstats
"data/$subject/pet/${subject}_PiB_5070_MNI.nii.gz" \
                    -k vois/voi_cereb_binary.nii.gz -M
2>/dev/null)
        if [ -n "$cerebellar" ]; then
            if (( $(echo "$cerebellar > 5" | bc -l 2>/dev/null)
)); then
                echo "‰ $subject: High cerebellar ($cerebellar) -
Applying normalization"
            fi
        fi
    fi
done
```

3.2 Step 2: Apply Intensity Normalization

```
# Normalize PET files with inconsistent scaling
for subject in AD* YC*; do
    PET_FILE="data/$subject/pet/${subject}_PiB_5070_MNI.nii.gz"
    if [ -f "$PET_FILE" ]; then
        cerebellar=$(fslstats "$PET_FILE" -k
vois/voi_cereb_binary.nii.gz -M 2>/dev/null)

        if [ -n "$cerebellar" ] && (( $(echo "$cerebellar > 5" |
bc -l 2>/dev/null) )); then
            echo "Normalizing $subject: cerebellar = $cerebellar"
            scale=$(echo "$REF_CEREBELLAR / $cerebellar" | bc -l
2>/dev/null)
            fslmaths "$PET_FILE" -mul $scale \

"data/$subject/pet/${subject}_PiB_5070_MNI_norm.nii.gz"
        fi
    fi
done
```

4 VOI (Volume of Interest) Processing

4.1 Step 1: Convert probability maps to binary masks (threshold at 0.5)

```
fslmaths vois/voi_ctx_2mm.nii -thr 0.5 -bin
vois/voi_ctx_binary.nii.gz
fslmaths vois/voi_CerebGry_2mm.nii -thr 0.5 -bin
vois/voi_cereb_binary.nii.gz
```

4.2 Step 2: Align VOIs to subject's PET space

```
flirt -in ${MNI_TEMPLATE} \
      -ref data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_MNI_thr.nii.gz
\
      -out vois/MNI_to_${SUBJECT}.nii.gz \
      -omat vois/MNI_to_${SUBJECT}.mat \
      -dof 6


flirt -in vois/voi_ctx_binary.nii \
      -ref data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_MNI_thr.nii.gz
\
      -out vois/voi_ctx_${SUBJECT}.nii \
      -applyxfm -init vois/MNI_to_${SUBJECT}.mat \
      -interp nearestneighbour


flirt -in vois/voi_cereb_binary.nii \
      -ref data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_MNI_thr.nii.gz
\
      -out vois/voi_cereb_${SUBJECT}.nii \
      -applyxfm -init vois/MNI_to_${SUBJECT}.mat \
      -interp nearestneighbour
```

5 SUVR Calculation

5.1 Step 1: Extract regional mean values

```
CORTICAL=$(fslstats
data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_MNI_thr.nii.gz \
          -k vois/voi_ctx_${SUBJECT}.nii -M)


CEREBELLAR=$(fslstats
data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_MNI_thr.nii.gz \
            -k vois/voi_cereb_${SUBJECT}.nii -M)
```

5.2 Step 2: Calculate SUVR with quality check

```
SUVR=$(echo "${CORTICAL} / ${CEREBELLAR}" | bc -l 2>/dev/null)


# Quality check
if [ -n "$SUVR" ]; then
    if (( $(echo "$CEREBELLAR > 5" | bc -l 2>/dev/null) )); then
        echo "WARNING: High cerebellar value ($CEREBELLAR) -
consider normalization"
    fi

    if [ "$GROUP" = "AD" ] && (( $(echo "$SUVR < 1.4" | bc -l
2>/dev/null) )); then
        echo "CHECK: AD subject with low SUVR ($SUVR)"
    fi
fi


echo "Subject: ${SUBJECT}"
echo "Cortical mean: ${CORTICAL}"
echo "Cerebellar mean: ${CEREBELLAR}"
echo "SUVR: ${SUVR}"
```

5.3 Step 3: Output results

```
echo "Subject: ${SUBJECT}"
echo "Cortical mean: ${CORTICAL}"
echo "Cerebellar mean: ${CEREBELLAR}"
echo "SUVR: ${SUVR}"
```

6 Batch Processing Script

6.1 Create batch processing script

```
#!/bin/bash
# Create batch processing script
MNI_TEMPLATE="YOUR_MNI_PATH_HERE"
OUTPUT="results_$(date +%Y%m%d).csv"

echo "Subject,Group,Cortical,Cerebellar,SUVR,QC_Status" >
"$OUTPUT"

for SUBJECT in AD01 AD02 AD03 AD04 AD05 AD06 AD07 AD08 AD09 AD10 \
               YC101 YC102 YC103 YC104 YC105 YC106 YC107 YC108
YC109 YC110; do

    echo "Processing ${SUBJECT}..."

    # Determine group
    if [[ "$SUBJECT" == AD* ]]; then
        GROUP="AD"
    else
        GROUP="YC"
    fi

    # Find best PET file (normalized first)
    PET_FILE=""
    for file in
"data/$SUBJECT/pet/${SUBJECT}_PiB_5070_MNI_norm.nii.gz" \
                "data/$SUBJECT/pet/${SUBJECT}_PiB_5070_MNI.nii.gz"
\

"data/$SUBJECT/pet/${SUBJECT}_PiB_5070_MNI_thr.nii.gz"; do
        if [ -f "$file" ]; then
            PET_FILE="$file"
            break
        fi
    done

    if [ -z "$PET_FILE" ]; then
        echo "$SUBJECT,$GROUP,,,NO_PET_FILE" >> "$OUTPUT"
        continue
    fi

    # Direct extraction
    CORTICAL=$(fslstats "$PET_FILE" -k vois/voi_ctx_binary.nii.gz
-M 2>/dev/null)
    CEREBELLAR=$(fslstats "$PET_FILE" -k
```

```
vois/voi_cereb_binary.nii.gz -M 2>/dev/null)

    if [ -z "$CORTICAL" ] || [ -z "$CEREBELLAR" ]; then
        echo
"$SUBJECT,$GROUP,$CORTICAL,$CEREBELLAR,,EXTRACTION_FAILED" >>
"$OUTPUT"
        continue
    fi

    SUVR=$(echo "$CORTICAL / $CEREBELLAR" | bc -l 2>/dev/null)

    # QC Status
    QC="PASS"
    if (( $(echo "$CEREBELLAR > 5" | bc -l 2>/dev/null) )); then
        QC="HIGH_CEREB"
    fi
    if [ "$GROUP" = "AD" ] && (( $(echo "$SUVR < 1.4" | bc -l
2>/dev/null) )); then
        QC="AD_LOW_SUVR"
    fi

    echo "$SUBJECT,$GROUP,$CORTICAL,$CEREBELLAR,$SUVR,$QC" >>
"$OUTPUT"
    echo "  SUVR: $SUVR ($QC)"
done

echo "Results saved to: $OUTPUT"
```

7 Quality Control

7.1 Step 1: Validation Check

```
# Validate with published AD01 value (should be 2.524)
AD01_SUVR=2.459  # Your result
PUBLISHED_SUVR=2.524


DIFF_PCT=$(echo "scale=2; (${AD01_SUVR} -
${PUBLISHED_SUVR})/${PUBLISHED_SUVR}*100" | bc -l)
echo "AD01 SUVR: ${AD01_SUVR} (published: ${PUBLISHED_SUVR})"
echo "Difference: ${DIFF_PCT}% (should be <5%)"
```

7.2 Step 2: Data Quality Checks

```
# Check cerebellar values (should be 1-4 for PiB)
if (( $(echo "${CEREBELLAR} < 1 || ${CEREBELLAR} > 10" | bc -l)
)); then
    echo "WARNING: Unusual cerebellar value (${CEREBELLAR})"
    echo "Check VOI alignment and PET normalization"
fi


# Check PET value ranges
fslstats data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_MNI_thr.nii.gz -R
-M -S
# Expected: mean ~4-10, max ~20-30 for PiB
```

7.3 Step 3: Visual Inspection

```
# Visual check of alignment
fsleyes ${MNI_TEMPLATE} \
       data/${SUBJECT}/pet/${SUBJECT}_PiB_5070_MNI_thr.nii.gz -cm
hot \
       vois/voi_cereb_${SUBJECT}.nii -cm green -a 50 &
```

8 Statistical Analysis

8.1 Step 1: Group Comparison

```
# After processing all subjects, calculate group statistics
echo "=== GROUP STATISTICS ==="


# AD group
echo "AD subjects:"
awk -F, '$2=="AD" {sum+=$4; count++} END{print "Mean SUVR:",
sum/count}' results.csv


# YC group
echo "Young controls:"
awk -F, '$2=="YC" {sum+=$4; count++} END{print "Mean SUVR:",
sum/count}' results.csv
```

8.2 Step 2: Centiloid Calculation

```
# Calculate Centiloid values
MEAN_YC=1.04  # Your YC group mean
MEAN_AD=2.53  # Your AD group mean


# For each subject: CL = 100 × (SUVr - mean_YC) / (mean_AD -
mean_YC)
CL=$(echo "scale=2; 100 * (${SUVR} - ${MEAN_YC}) / (${MEAN_AD} -
${MEAN_YC})" | bc -l)
echo "Centiloid value: ${CL}"
```

9 Troubleshooting

9.1 Common Issues and Solutions:

	Issue	Possible Cause	Solution
	Cerebellar values too	VOI misalignme	Check affine

	high (>10)	nt	registration, visualize alignment
	PET values extremely high (>100)	Intensity scaling issue	Check original PET units, may need scaling
	Registration fails	Poor image quality	Check BET extraction, adjust -f parameter
	SUVR ~1.0 for AD subjects	VOI misalignment	Ensure proper VOI alignment to PET space
	Memory errors	Large images	Use fslchfiletype to convert to NIFTI_GZ

Debugging Commands:

```
# Check image dimensions match
fslinfo image1.nii.gz | grep -E "dim[123]"
fslinfo image2.nii.gz | grep -E "dim[123]"


# Check orientation
fslhd image.nii.gz | grep -E "qform|sform"


# Quick visual check
fslview image1.nii.gz image2.nii.gz &
```

Intensity Scaling Issues (Common Problem)

Problem: SUVR values too low (<1.4) for AD subjects
Symptoms:
- Cerebellar mean > 5.0 (should be 1-4 for PiB)
- SUVR unexpectedly low even with good alignment

Solution:

```bash
# 1. Check cerebellar value
cerebellar=$(fslstats PET_FILE -k vois/voi_cereb_binary.nii.gz -M)
echo "Cerebellar value: $cerebellar"

# 2. If >5, normalize
if (( $(echo "$cerebellar > 5" | bc -l) )); then
    echo "Applying intensity normalization..."
    scale=$(echo "3.5 / $cerebellar" | bc -l)  # Target value
    fslmaths PET_FILE -mul $scale PET_FILE_normalized.nii.gz
    # Re-extract with normalized file
fi
```

Problem: Extraction returns extremely high values (>100)
Cause: PET file in raw counts instead of standardized units
Solution: Use intensity normalization as above

10 Expected Results

10.1 Actual Validation Against GAIN:

- AD01: Our SUVR = 2.18, GAIN = 2.52 (13% difference)
- AD05: Our SUVR = 2.32, GAIN = 2.54 (9% difference)
- AD20: Our SUVR = 2.50, GAIN = 2.45 (2% difference)
- AD Group (validated): ~2.3 ± 0.3 SUVR
- YC Group (expected): ~1.0 ± 0.2 SUVR

Quality Control Ranges:

- Cerebellar Gray (PiB): 1.0 - 4.0 SUV
- AD SUVR (CG reference): >1.4 (typically 1.8-3.0)
- YC SUVR (CG reference): <1.2 (typically 0.8-1.2)
- Acceptable GAIN difference: <20% for validated subjects

10.2 Validation Metrics

AD01 SUVR should be 2.524 ± 0.126 (within 5%)
Group separation: AD > YC ($p < 0.05$)
Cerebellar values consistent across subjects

10.3 Output Files

```
processed/
├── per_subject/
│   ├── ${SUBJECT}_PET_MNI_thr.nii.gz
│   ├── ${SUBJECT}_T1_MNI.nii.gz
│   └── ${SUBJECT}_VOIs_aligned.nii.gz
├── results/
│   ├── suvr_values.csv
│   ├── centiloid_values.csv
│   └── qc_report.txt
└── transforms/ (transformation matrices)
```

11 **Time Requirements**

11.1

	Step	Time per Subject	Notes
	T1 processing	1-2 minutes	BET + FLIRT
	PET processing	2-3 minutes	Coregistration + normalization
	VOI processing	1 minute	Alignment
	SUVR calculation	<30 seconds	Value extraction
	Total	~5 minutes	Per subject

Total for 10 subjects: ~50 minutes
Total for full dataset (59 subjects): ~5 hours

FSL + SPM PIPELINE

12 **Step 1: Data Organization and DICOM Conversion:**
Duration:~5 minutes per subject

12.1 **Create Project Directory Structure**

```
mkdir -p ~/Desktop/derivatives
```

12.2 **Extract Subject Data**

Extract subject-specific data from bulk ZIP files:

```
unzip -n ~/Downloads/AD-100_MR.zip "*AD${subject_id}*" -d
~/Desktop/derivatives/

# PET data
unzip -n ~/Downloads/AD_PET_01-25.zip "*AD${subject_id}*" -d
~/Desktop/derivatives/
```

12.3 **Create BIDS Directory Structure**

```
mkdir -p ~/Desktop/derivatives/data/sub-${subject_id}/anat
mkdir -p ~/Desktop/derivatives/data/sub-${subject_id}/pet
```

12.4 **Convert DICOM to NIfTI**

Load the dcmniix module and convert:

```
ml dcm2niix/v1.0.20240202

# Convert MRI
dcm2niix -o data/sub-${subject_id}/anat -f "sub-${subject_id}_T1w"
-z y -ba y -v y <MRI_DICOM_DIR>

# Convert PET
dcm2niix -o data/sub-${subject_id}/pet -f "sub-${subject_id}_pet"
-z y -ba y -v y <PET_DICOM_DIR>
```

12.5 **Merge Split PET Series (if applicable)**

If PET data is split across multiple series:

```
ml fsl/6.0.7.8
fslmerge -t sub-${subject_id}_pet.nii.gz <part1.nii.gz>
<part2.nii.gz> ...
```

Expected Outputs:

```
data/sub-XX/anat/sub-XX_T1w.nii.gz
data/sub-XX/pet/sub-XX_pet.nii.gz
```

13 **Step 2: PET Preprocessing (Motion Correction & Spatial Normalization)**
Duration: ~15-20 minutes per subject

13.1 **Load Required Software**

```
ml fsl/6.0.7.8
ml spm12/r7771
```

13.2 **Extract Static PET Frames**
Read frame range from framing_info.csv (typically 50-70 min post-injection):

```
cd ~/Desktop/derivatives/data/sub-${subject_id}/pet/

# Split 4D PET into individual frames
fslsplit sub-${subject_id}_pet.nii.gz frames/vol_

# Decompress frames for SPM
gunzip -f frames/vol_*.nii.gz
```

13.3 **Motion Correction (SPM Realignment)**
Run SPM realignment to correct for head motion and generate mean image:
SPM Parameters:

- Quality: 0.9
- Separation: 4 mm

- FWHM: 5 mm
- Register to mean: Yes
- Interpolation: 2nd degree B-spline

```
% SPM Batch
matlabbatch{1}.spm.spatial.realign.estwrite.eoptions.quality =
0.9;
matlabbatch{1}.spm.spatial.realign.estwrite.eoptions.sep = 4;
matlabbatch{1}.spm.spatial.realign.estwrite.eoptions.fwhm = 5;
matlabbatch{1}.spm.spatial.realign.estwrite.roptions.which = [2
1]; % Mean image
```

Output: meanvol_XXXX.nii renamed to sub-XX_pet_avg.nii

13.4 **Reorient to Standard Space**

```
fslreorient2std sub-${subject_id}_pet_avg.nii
sub-${subject_id}_pet_avg_reoriented.nii.gz
gunzip -f sub-${subject_id}_pet_avg_reoriented.nii.gz
```

13.5 **Prepare MRI**

```
cd ~/Desktop/derivatives/data/sub-${subject_id}/anat/
cp sub-${subject_id}_T1w.nii.gz
sub-${subject_id}_T1w_preproc.nii.gz
fslreorient2std sub-${subject_id}_T1w_preproc.nii.gz
sub-${subject_id}_T1w.reoriented.nii.gz
gunzip -f sub-${subject_id}_T1w.reoriented.nii.gz
```

13.6 **Origin Centering**

Reset image origins to geometric center for improved coregistration:

```
% Calculate geometric center
v = spm_vol(filename);
com = (v.dim(1:3)' + 1) / 2;
R = v.mat(1:3, 1:3);
T_new = -R * com;
M_new = [R T_new; 0 0 0 1];
spm_get_space(filename, M_new);
```

13.7 **Coregister PET to MRI**

SPM Coregistration Parameters:

- Cost function: Normalized Mutual Information (NMI)
- Separation: [4 2] mm
- Tolerances: [0.02 0.02 0.02 0.001 0.001 0.001 0.01 0.01 0.01 0.001 0.001 0.001]
- FWHM: [7 7] mm

```
matlabbatch{1}.spm.spatial.coreg.estimate.ref = {'T1w.nii,1'};
matlabbatch{1}.spm.spatial.coreg.estimate.source =
{'pet_avg.nii,1'};
matlabbatch{1}.spm.spatial.coreg.estimate.eoptions.cost_fun =
'nmi';
```

13.8 **Unified Segmentation (MRI)**

Segment MRI and generate deformation fields for MNI normalization:

```
matlabbatch{2}.spm.spatial.preproc.channel.vols = {'T1w.nii,1'};
matlabbatch{2}.spm.spatial.preproc.warp.affreg = 'mni';
matlabbatch{2}.spm.spatial.preproc.warp.write = [1 1]; %
Deformation fields
```

Output: y_sub_XX_T1w.reoriented.nii (forward deformation field)

13.9 **Normalize PET to MNI Space**

Apply the deformation field from MRI segmentation to PET:

Normalization Parameters:

- Bounding box: [-90 -126 -72; 91 91 109] mm
- Voxel size: [2 2 2] mm isotropic
- Interpolation: 4th degree B-spline

```
matlabbatch{3}.spm.spatial.normalise.write.woptions.bb = [-90 -126
-72; 91 91 109];
matlabbatch{3}.spm.spatial.normalise.write.woptions.vox = [2 2 2];
```

Output:wsub-XX_pet_avg.nii

13.10 **Gaussian Smoothing**

Apply 8mm FWHM Gaussian smoothing (Centiloid standard for PiB):

```
matlabbatch{4}.spm.spatial.smooth.fwhm = [8 8 8];
```

Output:swsub-XX_pet_avg.nii gzipped to sub-XX_pet_to_MNI_smoothed.nii.gz

14 **Step 3: Quality Control**

Duration:~1 minute per subject

14.1 **Setup and Load FSL**

```
ml fsl/6.0.7.8

# Define paths
BASE_DIR=~/Desktop/derivatives
SUB_DIR="${BASE_DIR}/data/sub-${subject_id}"
QC_DIR="${BASE_DIR}/QC/sub-${subject_id}"
mkdir -p "$QC_DIR"

# Define image paths
T1="${SUB_DIR}/anat/sub-${subject_id}_T1w.reoriented.nii"
PET_MNI="${SUB_DIR}/pet/sub-${subject_id}_pet_to_MNI_smoothed.nii.
gz"
AVG_PET="${SUB_DIR}/pet/sub-${subject_id}_pet_avg.nii"
MNI_TEMPLATE="${FSLDIR}/data/standard/MNI152_T1_2mm_brain.nii.gz"

# Centiloid masks
MASK_CEREB="data/references/centiloid_masks/voi_WhlCbl_2mm.nii"
MASK_CTX="data/references/centiloid_masks/voi_ctx_2mm.nii"
```

14.2 **Coregistration Check**

Overlay averaged PET onto native T1 MRI to verify PET-MRI alignment:

```
fsleyes render --outfile "${QC_DIR}/qc_coreg.png" \
    --size 1200 400 \
    --scene ortho \
    "$T1" --overlayType volume --name "T1" \
    "$AVG_PET" --overlayType volume --name "PET" --cmap hot --
alpha 50
```

Expected result: PET uptake should align with brain anatomy in MRI.

14.3 **Normalization Check**

Overlay MNI-normalized PET onto MNI152 template to verify spatial normalization:

```
fsleyes render --outfile "${QC_DIR}/qc_norm.png" \
    --size 1200 400 \
    --scene ortho \
    "$MNI_TEMPLATE" --overlayType volume --name "MNI152" \
    "$PET_MNI" --overlayType volume --name "PET_MNI" --cmap hot --
alpha 50
```

Expected result: Brain boundaries should match MNI template.

14.4 **Mask Alignment Check**

Overlay Centiloid masks onto normalized PET to verify ROI placement:

```
fsleyes render --outfile "${QC_DIR}/qc_masks.png" \
    --size 1200 400 \
    --scene ortho \
    "$PET_MNI" --overlayType volume --name "PET_MNI" --cmap gray \
    "$MASK_CEREB" --overlayType volume --name "Cerebellum" --cmap
Blue --alpha 40 \
    "$MASK_CTX" --overlayType volume --name "Cortex" --cmap Red --
alpha 40
```

Expected result:

- Blue (Cerebellum) mask should cover cerebellar region
- Red (Cortex) mask should cover cortical gray matter

14.5 **Review Checklist**

	A	B
	Check	Pass Criteria
	Coregistration	PET aligns with MRI brain boundaries
	Normalization	Brain shape matches MNI template
	Mask alignment	Cerebellum mask in posterior fossa, cortex mask on gray matter

Expected Outputs:

- QC/sub-XX/qc_coreg.png -Coregistration overlay
- QC/sub-XX/qc_norm.png-Normalization overlay
- QC/sub-XX/qc_masks.png-Mask alignment overlay

15 **Step 4: SUVR and Centiloid Calculation**

Duration: ~2minutes per subject

15.1 **Load Required Software**

```
ml fsl/6.0.7.8
ml freesurfer/7.3.2
```

15.2 **Define Centiloid Parameters**

For [11C]PiB with whole cerebellum reference:

```
CENTILOID_SUVR_ZERO=1.009   # Young control mean SUVR
CENTILOID_SUVR_100=2.076    # Typical AD mean SUVR
```

15.3 **Resample PET to Mask Space**

Ensure PET and masks have identical geometry:

```
flirt -in sub-XX_pet_to_MNI_smoothed.nii.gz \
      -ref voi_WhlCbl_2mm.nii \
      -applyxfm -usesqform \
      -out sub-XX_pet_resampled_to_mask.nii.gz
```

15.4 **Extract Regional Mean Values**

```
# Reference region (Whole Cerebellum)
ref_mean=$(fslstats sub-XX_pet_resampled.nii.gz -k
voi_WhlCbl_2mm.nii -M)

# Target region (Global Cortex)
target_mean=$(fslstats sub-XX_pet_resampled.nii.gz -k
voi_ctx_2mm.nii -M)
```

15.5 **Calculate SUVR**

```
suvr=$(echo "scale=6; $target_mean / $ref_mean" | bc -l)
```

15.6 **Convert to Centiloid Scale**

Using the linear transformation formula:

```
Centiloid = 100 × (SUVR - SUVR_YC) / (SUVR_AD - SUVR_YC)
```

```
centiloid=$(echo "scale=4; 100 * ($suvr - 1.009) / (2.076 -
1.009)" | bc -l)
```

15.7 **Save Results**

Append to CSV file:

```
echo "$subject,$ref_mean,$target_mean,$suvr,$centiloid_value" >>
"$output_csv"
```

16 **Step 5: Statistical Validation**
Duration: ~1 minute

16.1 **Required Python Packages**

```
pip install pandas matplotlib scipy numpy
```

16.2 **Statistical Validation Script**
Save as statistical_test.py:

```
#!/usr/bin/env python3
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as stats
import numpy as np
import os
import sys

def main():
    # --- 1. CONFIGURE PATHS ---
    script_dir = os.path.dirname(os.path.abspath(__file__))
    project_root = os.path.dirname(os.path.dirname(script_dir))

    # Input files
    if len(sys.argv) > 1:
        results_file = sys.argv[1]
    else:
        results_file = os.path.join(project_root, "results",
"tables", "all_subjects_results.csv")

    ref_file = os.path.join(project_root, "data", "references",
"centiloid_values.csv")
    output_dir = os.path.join(project_root, "results", "reports")

    os.makedirs(output_dir, exist_ok=True)

    # --- 2. LOAD DATA ---
    print(f"Reading results from: {results_file}")
    df_calc = pd.read_csv(results_file)

    print(f"Reading reference from: {ref_file}")
    df_ref = pd.read_csv(ref_file)
    df_ref.columns = [c.strip() for c in df_ref.columns]

    print("Calculated Data Preview:")
    print(df_calc.head())

    # --- 3. MERGE DATASETS ---
    df_calc['subject_id'] =
df_calc['subject_id'].astype(str).str.strip()
    df_ref['Subject'] = df_ref['Subject'].astype(str).str.strip()

    df_merged = pd.merge(df_calc, df_ref, left_on='subject_id',
right_on='Subject', how='inner')
```

```
    if df_merged.empty:
        print("ERROR: No matching subjects found.")
        sys.exit(1)

    print(f"Successfully merged {len(df_merged)} subjects.")

    # --- 4. CORRELATION ANALYSIS ---
    # SUVR Correlation
    x_suvr = df_merged['global_cortical_suvr']
    y_suvr = df_merged['SUVR']
    r_suvr, p_suvr = stats.pearsonr(x_suvr, y_suvr)

    # Centiloid Correlation
    x_cl = df_merged['global_cortical_centiloid']
    y_cl = df_merged['Centiloid']
    r_cl, p_cl = stats.pearsonr(x_cl, y_cl)

    # --- 5. PRINT RESULTS ---
    print("\n=== STATISTICAL ANALYSIS RESULTS ===")
    print(f"Number of Subjects: {len(df_merged)}")
    print(f"\n1. SUVR Correlation:")
    print(f"   Pearson r: {r_suvr:.4f}")
    print(f"   p-value:   {p_suvr:.4e}")
    print(f"\n2. Centiloid Correlation:")
    print(f"   Pearson r: {r_cl:.4f}")
    print(f"   p-value:   {p_cl:.4e}")
    print("====================================\n")

    # --- 6. GENERATE PLOTS ---
    fig, axes = plt.subplots(1, 2, figsize=(12, 5))

    # Plot 1: SUVR
    axes[0].scatter(x_suvr, y_suvr, alpha=0.7)
    axes[0].set_title(f'SUVR Correlation\nr={r_suvr:.3f}, p=
{p_suvr:.3e}')
    axes[0].set_xlabel('Calculated SUVR')
    axes[0].set_ylabel('Reference SUVR')
    m, b = np.polyfit(x_suvr, y_suvr, 1)
    axes[0].plot(x_suvr, m*x_suvr + b, color='red', linestyle='--
')
    axes[0].grid(True, linestyle=':', alpha=0.6)

    # Plot 2: Centiloid
    axes[1].scatter(x_cl, y_cl, color='green', alpha=0.7)
```

```
    axes[1].set_title(f'Centiloid Correlation\nr={r_cl:.3f}, p=
{p_cl:.3e}')
    axes[1].set_xlabel('Calculated Centiloid')
    axes[1].set_ylabel('Reference Centiloid')
    m, b = np.polyfit(x_cl, y_cl, 1)
    axes[1].plot(x_cl, m*x_cl + b, color='red', linestyle='--')
    axes[1].grid(True, linestyle=':', alpha=0.6)

    plt.tight_layout()
    output_plot = os.path.join(output_dir,
"correlation_plots.png")
    plt.savefig(output_plot)
    print(f"Plots saved to: {output_plot}")

if __name__ == "__main__":
    main()
```

16.3 **Run Statistical Validation**

python3 statistical_test.py results/tables/all_subjects_results.csv

16.4 **Expected Output**

```
=== STATISTICAL ANALYSIS RESULTS ===
Number of Subjects: 25

1. SUVR Correlation:
   Pearson r: 0.9700
   p-value:   1.23e-15

2. Centiloid Correlation:
   Pearson r: 0.9700
   p-value:   1.23e-15
```

Batch Processing

17 **Create Batch Script**

save as 00_batch_master.sh

```
#!/bin/bash
# Master Batch Processing Script for PET-NeuroProject
set -e

# Define subject range (default: 1-25)
start_sub=${1:-1}
end_sub=${2:-25}

# Determine paths
SCRIPT_DIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" &> /dev/null
&& pwd )"
PROJECT_ROOT="$(dirname "$(dirname "$SCRIPT_DIR")")"

# Output files
log_file="batch_processing_log_sub${start_sub}-${end_sub}_$(date
+%Y%m%d_%H%M%S).txt"
master_csv="${PROJECT_ROOT}/results/tables/all_subjects_results.cs
v"

mkdir -p "$(dirname "$master_csv")"

# Reset master CSV if starting from subject 1
if [ "$start_sub" -eq 1 ] && [ -f "$master_csv" ]; then
    rm "$master_csv"
fi

echo "Starting Batch Processing: subjects $start_sub to $end_sub"
| tee -a "$log_file"

for ((i=start_sub; i<=end_sub; i++)); do
    subject_id=$(printf "%02d" $i)

    echo "================================================" |
tee -a "$log_file"
    echo "Processing Subject: $subject_id" | tee -a "$log_file"
    echo "================================================" |
tee -a "$log_file"

    # Step 1: Data Organization
    echo "Running 01_data_org.sh..." | tee -a "$log_file"
    "${SCRIPT_DIR}/01_data_org.sh" "$subject_id" >> "$log_file"
2>&1
    if [ $? -ne 0 ]; then
```

```
        echo "ERROR: Data organization failed for $subject_id" |
tee -a "$log_file"
        continue
    fi

    # Step 2: Preprocessing
    echo "Running 03_preprocessing.sh..." | tee -a "$log_file"
    "${SCRIPT_DIR}/03_preprocessing.sh" "$subject_id" >>
"$log_file" 2>&1
    if [ $? -ne 0 ]; then
        echo "ERROR: Preprocessing failed for $subject_id" | tee -
a "$log_file"
        continue
    fi

    # Step 3: Visual QC
    echo "Running visual_qc.sh..." | tee -a "$log_file"
    "${SCRIPT_DIR}/../qc/visual_qc.sh" "$subject_id" >>
"$log_file" 2>&1

    # Step 4: Analysis
    echo "Running analysis.sh..." | tee -a "$log_file"
    "${SCRIPT_DIR}/../Analysis/analysis.sh" "$subject_id"
"$master_csv" >> "$log_file" 2>&1

    echo "Finished Subject $subject_id at $(date)" | tee -a
"$log_file"
done

# Generate QC Report
echo "Generating HTML QC Report..." | tee -a "$log_file"
python3 "${SCRIPT_DIR}/../qc/generate_report.py" >> "$log_file"
2>&1

echo "Batch Processing Complete. Results: $master_csv"
```

17.1 Running the Batch Script

```
# Process all subjects (01-25)
./00_batch_master.sh 1 25

# Process specific range
./00_batch_master.sh 5 10    # Only subjects 05-10

# Resume from a specific subject
./00_batch_master.sh 15 25   # Subjects 15-25
```

17.2 Batch Script Workflow

```
For each subject (01 to 25):
├── 01_data_org.sh      → Extract & convert DICOM to NIfTI
├── 03_preprocessing.sh → Motion correction, normalization,
smoothing
├── visual_qc.sh        → Generate QC images
└── analysis.sh         → Calculate SUVR & Centiloid

After all subjects:
└── generate_report.py  → Aggregate QC report
```

17.3 Monitoring Progress

```
# Watch log file in real-time
tail -f batch_processing_log_*.txt

# Check disk usage during processing
df -h
```

17.4 Expected Results

Quality Metrics

- **Correlation (r):** ≥ 0.95 between calculated and reference values
- **Processing time:** ~20-30 minutes per subject (batch mode)

18 Troubleshooting

Common issues

1. **Coregistration failure:** Check image orientations with fslorient -getorient
2. **Low correlation values:** Review QC images for specific subjects
3. **Missing frames:** Verify frame range in framing_info.csv
4. **Disk space errors:** Enable cleanup in preprocessing script

19 **Note**

FSL Pipeline: Results from processing subjects 5, 9, 10 and 13 were outliers
Hybrid FSL/SPM Pipeline: Results from subjects 5, 9, and 10 were outliers

Outliers were excluded from the analysis.

20 **Support**

For questions or issues:

- GitHub Issues: https://github.com/HolaDav/FLEX-PROJECT-PET-TEAM-O/issues
- Email: oladejidavid12@email.com
  adeniyiabdullahi30@gmail.com

Protocol references

1. FSL- Jenkinson, M., Beckmann, C. F., Behrens, T. E., Woolrich, M. W. & Smith, S. M. FSL. *Neuroimage* **62**, 782–790 (2012). https://doi.org/10.1016/j.neuroimage.2011.09.015

2. McCarthy, P. *FSLeyes* version 1.7.0 (Zenodo, 2023). https://doi.org/10.5281/zenodo.7038115

3. Jenkinson, M. & Smith, S. A global optimisation method for robust affine registration of brain images. *Med. Image Anal.* **5**, 143–156 (2001). https://doi.org/10.1016/S1361-8415(01)00036-6

4. Friston, K. J. *et al. Statistical Parametric Mapping: The Analysis of Functional Brain Images* (Elsevier, 2007). **https://www.fil.ion.ucl.ac.uk/spm/doc/books/hbm1/**

5. Ashburner, J. & Friston, K. J. Unified segmentation. *Neuroimage* **26**, 839–851 (2005). https://doi.org/10.1016/j.neuroimage.2005.02.018

6. Van Rossum, G. & Drake, F. L. *Python 3 Reference Manual* (CreateSpace, 2009). **https://docs.python.org/3/reference/**

7. Klunk, W. E. *et al.* The Centiloid Project: standardising quantitative amyloid plaque estimation by PET. *Alzheimers Dement.* **11**, 1–14 (2015). https://doi.org/10.1016/j.jalz.2014.07.003

8. Renton, A. I. *et al.* Neurodesk: an accessible, flexible and portable data analysis environment for reproducible neuroimaging. *Nat. Methods* **21**, 804–808 (2024). https://doi.org/10.1038/s41592-023-02145-x