1. Introduction

1.1. Discrete classification algorithm for Limbic and Hippocampal Sparing:

We have measures that we will refer to as the following for simplicity:

- 2 hippocampal measures: H1, H2
- 3 cortical measures: C1, C2, C3

From these we derive three more quantities:

- Hippocampal mean: H3 = (H1 + H2) / 2
- Cortical mean: C4 = (C1 + C2 + C3) / 3
- Ratio of Hippocampal to Cortical Means: R = H3 / C4

For classification of an AD case as the Hippocampal Sparing subtype we need the following, where the medians and percentiles are relative to a reference set:

- R < 25th percentile,
- All of H1, H2, H3 < their medians
- At least 3 of C1, C2, C3, C4 ≥ medians

For classification of a case as Limbic we need:

- R ≥ 75th percentile,
- At least 2 of H1, H2, H3 ≥ their medians,
- At least 3 of C1, C2, C3, C4 ≤ medians.

Table 1. Percentile values in algorithm using Original reference set and FLAME.

Measure	Percentile	Original	FLAME
R	25	1.11	1.0855
	75	3.6	3.75
H1	50	12	12
H2	50	20	20
H3	50	17.5	16.5
C1	50	10.5	10
C2	50	8	7
C3	50	5	5
C4	50	8.667	8

Note: the 50th percentile is the median.

Note that due to the discreteness of the distributions (H1, H2, C1, C2, C3 are counts), the use of '<' versus '≤', or '≥' versus' >', as well as rounding (e.g. of the median of C4), can make a difference to the final classification for some cases.

In the published version of the algorithm the percentiles were identified from a group of ~800 AD cases. From this point forward we will use the FLAME cohort as our reference set of AD cases. The benchmark percentiles are very similar as expected (Table 1).

1.2. The continuous score:

In the following sections we define a continuous **Hippocampal Score** taking values in [0,40] that corresponds exactly to the algorithm above in that the score indicates the AD subtype:

- **[0,10) → Hippocampal Sparing**
- **[10,30) → Typical AD**
- **[30,40] → Limbic**

2. Creation of the continuous score:

2.1. The algorithm can be made more or less stringent:

We first note that we could make the algorithm more stringent by changing the reference percentiles, e.g. a more stringent version of classification of Hippocampal Sparing might use the 15^{th} rather than 25^{th} percentile of R, the 30^{th} percentiles of H1, H2, H3 and the 70^{th} percentiles of C1, C2, C3, C4:

- R < 15^{th} percentile,
- **All** of H1, H2, H3 < their 30^{th} percentiles
- **At least 3** of C1, C2, C3, C4 ≥ 70^{th} percentiles

A more extreme Limbic classification as Limbic might require:

- R ≥ 90^{th} percentile of R,
- **At least 2** of H1, H2, H3 ≥ their 80^{th} percentiles,
- **At least 3** of C1, C2, C3, C4 ≤ their 20^{th} percentiles.

2.2. The algorithm can be generalized to correspond to any percentile k of R:

We can re-express the algorithm for any k in [0, 50) or (50, 100]:

For **k in [0, 50)** an AD case is **Hippocampal Sparing** if:

- R < k^{th} percentile of R, and:
- **All** of H1, H2, H3 < their $(2k)^{th}$ percentiles, and:
- **At least 3** of C1, C2, C3, C4 ≥ $(100 – 2k)^{th}$ percentiles.

For **k in (50, 100]** an AD case is **Limbic** if:

- R ≥ k^{th} percentile of R,
- **At least 2** of H1, H2, H3 ≥ their $(100\text{-}2(100\text{-}k))^{th}$ percentiles,
- **At least 3** of C1, C2, C3, C4 ≤ their $(2(100\text{-}k))^{th}$ percentiles.

2.3. Using the generalized algorithm to create a score:

The score k* is defined as the most extreme value of k (with respect to distance from 50) such that the Hippocampal criteria, or the Limbic criteria hold.

Alternatively stated, we can assign a score k* taking a value in [0, 100] to any AD case as follows:

- if R < median: k* = min { k: Hippocampal Sparing criteria met }
- if R > median: k* = max { k: Limbic criteria met }
- If R = median: k* = 50

The subtype of AD is then classified as according to the value of k*

- [0,25) → Hippocampal Sparing
- [25,75) → Typical AD
- [75,100] → Limbic

To prevent inaccurate interpretation of k* as a percentage or probability, we scale to create our final Hippocampal Sparing score as 0.4 × k* taking values in [0,40] with interpretation as in section 1.

3. Re-expression of the scoring algorithm to facilitate computation:

To describe the CLix algorithm briefly. For a new case, we have H1* and H2* to represent the neurofibrillary tangle (NFT) counts in the hippocampal subsectors of CA1 and Subiculum; C1*, C2* and C3* to represent the NFT counts in the cortical regions of Temporal, Parietal and Frontal.

Hippocampal Mean: H3* = (H1* + H2*) / 2
Cortical Mean: C4* = (C1* + C2* + C3*) / 3
Ratio of Hippocampal to Cortical Means: R* = H3* / C4*

To assign a CLix score, this R* is then measured in terms of the distribution of R from the reference dataset resulting in kR. kR is the proportion of the reference dataset that is less than R*. We handle kR one of the following three ways. If kR is equal to 50, we multiply by 0.4 in order to rescale and account for kR being a proportion and the CLix score is calculated as 0.4 * kR.

If kR is less than 50, the CLix score will be 0.4 times the maximum of kR, kH_L, and kC_L. kH_L is the largest of the proportion of Hi in the reference dataset that is less than Hi* divided by 2, with i = 1, 2, or 3. kC_L is the second largest of the proportion of Cj in the reference dataset that is greater than or equal to Cj* divided by 2, with j = 1, 2, 3, or 4.

If kR is greater than 50, the CLix score will be 0.4 times the minimum of kR, kH_U, and kC_U. kH_U is the second smallest of the proportion of Hi in the reference dataset that is greater than or equal to Hi* divided by 2 and subtracted from 100, with i = 1, 2, or 3. kC_U is the second smallest of the proportion of Cj in the reference dataset that is less than or equal to Cj* divided by 2 and subtracted from 100, with j = 1, 2, 3, or 4.

For a new case i,

- **kR = PltR** where PltR is the proportion (%) of the of R in the reference distribution that is **less than** the R value of new case

- **If kR < 50**, let:
  - kH1 = PltH1 /2 with PltH1 the % of the reference distribution of H1 that is **less than** (<) H1 of new case
  - kH2 = PltH2 /2
  - kH3 = PltH3 /2
    - kH = max(kH1, kH2, kH3)

  - kC1 = PgeC1/2 with PgeC1 the % of the reference distribution of C1 that **is greater than or equal to** (≥) C1 of case
  - kC2 = PgeC2/2
  - kC3 = PgeC3/2
  - kC4 = PgeC4/2
    - kC = max2(kC1,kC2,kC3,kC4) with 'max2' denoting 2nd largest

      - **k* = max(kR, kH, kC)**

- **If kR>50** let:
    - kH1 = 100 - PgeH1/2 with PgeH1 the % of the reference distribution of H1 **greater than or equal to** (≥) H1 of new case
    - kH2 = 100 - PgeH2/2
    - kH3 = 100 - PgeH3/2
        - kH = min2(kH1, kH2, kH3) with 'min2' denoting 2nd smallest
    - kC1 = 100 - PleC1/2 with PleC1 the % of the reference distribution of C1 **less than or equal to** (≤) C1 of new case
    - kC2 = 100 - PleC2/2
    - kC3 = 100 - PleC3/2
    - kC4 = 100 - PleC4/2
        - kC = min2(kC1,kC2,kC3,kC4) with 'min2' denoting 2nd smallest
            - **k* = min(kR, kH, kC)**
- **If kR = 50**, set all the above quantities to 50 so:
    - **k* = 50**.

Note that k* can be considered as an adjusted or 'shrunken' version of kR in that it is shrunk towards the central value of 50. As above, the final Hippocampal Sparing score is defined as 0.4 × k*.

4. R Package

(1) Package name: `CLix`

- The `CLix` package returns the continuous measurement, corticolimbic index (`CLix.score`), which describes the severity of Alzheimer's disease (AD) and assigns an AD subtype (Hippocampal Sparing AD, Typical AD, or Limbic AD).

  `CLix.score` $\in [0, 10) \Longrightarrow$ Hippocampal Sparing AD (`HpSp`)
  `CLix.score` $\in [10, 30) \Longrightarrow$ Typical AD (`Typical`)
  `CLix.score` $\in [30, 40) \Longrightarrow$ Limbic AD (`Limbic`)

- The Corticolimbic Index creation algorithm and imputation for missing tangle counts values were described in detail within the attached files on GitHub (https://github.com/Translational-Neuropathology-Lab).

  The algorithm takes in a dataset, a reference dataset, and an identifier column. The `CLix.scoring` function then returns a CLix score (`CLix.score`), confidence interval [`score.lo, score.hi`], and subtype classification (`CLix.score.subtype`) for each case. The default reference dataset is the FLAME-AD cohort (n=1361).

- Installment
  `devtools::install_github("Translational-Neuropathology-Lab/CLix")`

(2) Function name: `CLix.scoring(…)`

- Inputs: a dataset, a reference dataset, and an identifier column
  Dataset must use column names of H1, H2, C1, C2, C3 to represent NFT counts in five key brain regions.
- Outputs: `CLix.score, score.lo, score.hi, CLix.score.subtype`
- Reference dataset: The FLAME (Florida Autopsied Multi-Ethnic) cohort (n=1361) is derived from the State of Florida brain bank housed at the Mayo Clinic Florida. Donors are derived through community engagement and participating Memory Disorder Clinics in the State of Florida's Alzheimer's Disease Initiative who may register for autopsy regardless of sex, race, or ethnicity.

(3) missingness imputation

- H1 and H2 refer to the neurofibrillary tangle (NFT) counts in the hippocampal subsectors of CA1 and Subiculum; C1, C2 and C3 refer to the NFT counts in the cortical regions of Temporal, Parietal and Frontal. To calculate the ratio of Hippocampal to Cortical means, R, so that to return the corticolimbic index (`CLix.score`), it requires non-missingness on these five measures.

- If all the Hs and Cs are missing, nothing will be returned from the function `CLix.scoring`. If not all the Hs and Cs are missing, the function `CLix.scoring` will return the imputed score `CLix.score` if the imputation criteria are met, the confidence interval [`score.lo, score.hi`] for `CLix.score`, and the assigned AD subtype `CLix.score.subtype`.
  The imputation criteria for `CLix.score` require at least one of two Hs and at least one of three Cs, and `CLix.score` was imputed by using user's own dataset. While the confidence interval [`score.lo, score.hi`] for `CLix.score` was imputed by using Florida Multiethnic Alzheimer's disease (FLAME-AD) dataset or user's own reference dataset.

- Imputation method:
  - `CLix.score`: If H1 is missing, using the value of non-missing H2 to replace H1, vice versa. If C1 is missing, using the average value of C2 and C3 (at least one of two Cs is non-missing) to replace C1, similarly for C2 and C3.

  - `score.lo`: (`refdat` refers to the FLAME-AD dataset or user's own reference dataset)
    H1 missing $\Longrightarrow$ replaced by max(0, min(H1 in `refdat`) – 1)
    H2 missing $\Longrightarrow$ replaced by max(0, min(H2 in `refdat`) – 1)
    C1 missing $\Longrightarrow$ replaced by max(C1 in `refdat`) + 1
    C2 missing $\Longrightarrow$ replaced by max(C2 in `refdat`) + 1
    C3 missing $\Longrightarrow$ replaced by max(C3 in `refdat`) + 1

  - `score.hi`:
    H1 missing $\Longrightarrow$ replaced by max(H1 in `refdat`) + 1
    H2 missing $\Longrightarrow$ replaced by max(H2 in `refdat`) + 1
    C1 missing $\Longrightarrow$ replaced by max(0, min(C1 in `refdat`) – 1)
    C2 missing $\Longrightarrow$ replaced by max(0, min(C2 in `refdat`) – 1)
    C3 missing $\Longrightarrow$ replaced by max(0, min(C3 in `refdat`) – 1)

(4) Reference

- Liesinger et. Al. Acta Neuropathologica 2018
  https://link.springer.com/article/10.1007/s00401-018-1908-x

- Santos et. al. Alzheimer's & Dementia 2019
  https://alz-journals.onlinelibrary.wiley.com/doi/epdf/10.1016/j.jalz.2018.12.013