**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

|  |  |  |  |
| --- | --- | --- | --- |
| **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. | | | |

* 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.

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 15th rather than 25th percentile of R, the 30th percentiles of H1, H2, H3 and the 70th percentiles of C1, C2, C3, C4:

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

A more extreme Limbic classification as Limbic might require:

* + - R ≥ 90th percentile of R,
    - **At least 2** of H1, H2, H3 ≥ their 80th percentiles,
    - **At least 3** of C1, C2, C3, C4 ≤ their 20th 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 < kth percentile of R, and:
    - **All** of H1, H2, H3 < their (2k)th percentiles, and:
    - **At least 3** of L1, L2, L3, L4 ≤ (2k)th percentiles.

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

* + - R ≥ kth percentile of R,
    - **At least 2** of H1, H2, H3 ≥ their (100-2(100-k))th percentiles,
    - **At least 3** of L1, L2, L3, L4 ≥ their (100-2(100-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:**

For a new case i, let:

* **kR = PltR** where PltR is the proportion (%) of the of R reference distribution that is **less than** the R value of new case
* **If kR < 50**, let:
  + kH1 = PltH1 /2 with PltH1 the % of the ref distn of H1 that is **<** H1 of case
  + kH2 = PltH2 /2
  + kH3 = PltH3 /2
    - kH = max(kH1, kH2, kH3)
  + kC1 = PgeC1/2 with PgeC1 the % of the ref distn **≥** 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 ref distn ≥ H1 of case
  + kH2 = 100 - PgeH2/2
  + kH3 = 100 - PgeH3/2
    - kH = min2(kH1, kH2, kH3)with ‘min2’ denoting ‘2nd largest’
  + kC1 = 100 - PleC1/2 with PleC1 the % of the ref distn ≤ C1 of case
  + kC2 = 100 - PleC2/2
  + kC3 = 100 - PleC3/2
  + kC4 = 100 - PleC4/2
    - kC = min2(kC1,kC2,kC3,kC4)
      * **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 Hippocampal Sparing AD (HpSp)

CLix.score Typical AD (Typical)

CLix.score Limbic AD (Limbic)

* The Corticolimbic Index creation algorithm and imputation for missing values are described in detail within the attached files on GitHubADD REFERENCE LINK. The algorithm takes in a dataset, a reference dataset, and 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("zpeng2020/ ")

(2) Function name: CLix.scoring(…)

* Inputs: a dataset, a reference dataset, and 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: FLAME-AD cohort represents the data from AD cases in the larger Florida Autopsied Multi-Ethnic series (n=1361), collected in 2015? more details?

(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.scorewas 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 replaced by max(0, min(H1 in refdat)– 1)

H2 missing replaced by max(0, min(H2 in refdat)– 1)

C1 missing replaced by max(C1 in refdat)+ 1

C2 missing replaced by max(C2 in refdat)+ 1

C3 missing replaced by max(C3 in refdat)+ 1

* score.hi:

H1 missing replaced by max(H1 in refdat)+ 1

H2 missing replaced by max(H2 in refdat)+ 1

C1 missing replaced by max(0, min(C1 in refdat)– 1)

C2 missing replaced by max(0, min(C2 in refdat)– 1)

C3 missing replaced by max(0, min(C3 in refdat)– 1)