U of A Applied Math Hackathon Challenge

Nick Henscheid, Quantitative Medicine Program, Critical Path Institute

March 1st 2025 DUE March 16th 2025 at Midnight

1 Introduction

Heterogeneity is a common factor in nearly every disease, and dealing with heterogeneity is the crux of clinical research and trial design. For many diseases, the biology underlying clinical heterogeneity is well-understood; cancer, for example, has benefited from decades of studies to understand genetic and environmental sources of heterogeneity and their impact on treatment response, and clinical care for cancer has improved greatly as a result. In many other diseases however, particularly rare neurodegenrative diseases such as Amyotrophic Lateral Sclerosis (ALS), heterogeneity is less well-understood. While genetic and biomarker-driven heterogeneity has been studied in ALS, our understanding of how these factors translate into clinical subtypes or progression patterns remains incomplete. One approach to possibly discovering clinically meaningful disease subtypes is to analyze data collected during studies, either clinical trials or natural history studies, to try and understand if subjects display a continuum of progression or can be grouped into discrete disease subtypes based on available data features, such as biomarkers or clinical outcomes.

The goal of this challenge is to explore a synthetic dataset that is representative of the type of data we analyze at C-Path. You will apply mathematical & machine learning modeling strategies to uncover possible underlying subtypes and better understand patterns of disease heterogeneity.

Notes:

1. You can use any programming language, but Python and R are recommended (industry standards)
2. You can use AI to help write code, format your results, ask questions, but think for yourself and approach this like you would a research problem.
3. While the data has been constructed to mimic some features of real ALS data, it is completely synthetic; *no conclusions about ALS should be drawn from this dataset!*

2 Data Description

For this challenge, you will be provided **TWO** `.csv` files with the structure outlined in Tables 1 and 2. The first, `train.csv` will be used to build a functional clustering method and a classification algorithm that predicts functional cluster from baseline covariates. The second, `test.csv` will contain **only** baseline covariates; you will then use the classifier that you built with the training data to predict the cluster number for the subjects in the test set.

These files, which are in 'tabular' form, contain data for a group of subjects. You can imagine that these data were recorded as part of a natural history study whereby subjects were followed without any intervention above and beyond 'standard-of-care': such studies are crucial for understanding diseases. Subjects are labeled with the integer index SID. For subject i in the training data, cognitive scores Y_{ij} are recorded for times t_{ij}. We also have 'baseline covariates', that is, for subject i we have a vector of features $\mathbf{x}_i$: we assume only Age at baseline (AGE), sex (SEX) and a baseline biomarker such as a lab test result (BIO). For the test set, you will only have baseline covariates and the baseline cognitive score (Y_{i0}). The variables, their types, min and max values are given in Table 3.

SID	T	Y	AGE	SEX	BIO
0	t_{00}	Y_{00}	x_{00}	x_{01}	x_{02}
0	t_{01}	Y_{01}	x_{00}	x_{01}	x_{02}
0	t_{02}	Y_{02}	x_{00}	x_{01}	x_{02}
1	t_{10}	Y_{10}	x_{10}	x_{11}	x_{12}
1	t_{11}	Y_{11}	x_{10}	x_{11}	x_{12}
1	t_{12}	Y_{12}	x_{10}	x_{11}	x_{12}
...	...	...	...	...	...

Table 1: Format of the `train.csv` file. Static ('baseline') variables are repeated. **You will return this file with an appended column** `cluster` that contains the cluster number predicted from your FDA approach.

SID	Y0	AGE	SEX	BIO
0	Y_{00}	x_{00}	x_{01}	x_{02}
1	Y_{10}	x_{10}	x_{11}	x_{12}
2	Y_{20}	x_{10}	x_{11}	x_{12}
...	...	...	...	...

Table 2: Format of the `test.csv` file. Only static ('baseline') variables are given - you must predict the cluster number from these variables. **You will return this file with an appended column** `pred_cluster` that contains the cluster number predicted from your classification algorithm.

Variable	Units	Type	Min	Max	Notes
SID	none	Int	0	N_{subj}	Subject ID
T	Years	Float	0	5	Times are not necessarily 'aligned' Subjects may have different number of times
Y	Score	Float	0	48	Arbitrary cognitive score (**higher = worse**)
AGE	Years	Int	0	100	Age at baseline ($t = 0$), **not** age at onset!
SEX	none	Binary	0	1	0 = Male, 1 = Female
BIO	none	Float	0	1	Arbitrary lab test units e.g. mg/dL

Table 3: Variable info

3 Example Python Envrionment

My recommendation for setting up your python environment would be the following, assuming you have anaconda installed. This will create an environment, install default packages, link your conda environment to the ipykernel (so you can access it in jupyter), then install the scikit-fda functional data analysis package and open a notebook.

```
conda create -n "hackathon_2025" python=3.9.0 anaconda
conda activate ALSFDA
python -m ipykernel install --user --name ALSFDA --display-name "Python (ALSFDA)"
pip install scikit-fda xgboost
jupyter notebook
```

Once your notebook is open, you can select the `Python (ALSFDA)` kernel and get started.

Note: If you decide to use R instead, you can check out the packages listed here:

https://cran.r-project.org/web/views/FunctionalData.html

4 Questions

There will be two main 'deliverables' for this project:

1. A code file with your work, either jupyter notebook (.ipynb), script (.py) or R notebook (.Rmd) or script (.R). I prefer notebooks as they are easier to run the 'chunks' to see your output, and you can write text blocks to provide some discussion e.g. 'here I tried XX method and it didn't work, so I changed it to YY method which seemed to work better'.

2. The two `.csv` files with the original data plus a new column called `cluster`. For `train.csv`, `cluster` will be the cluster label determined by your functional clustering method. For `test.csv`, `cluster` will be the cluster label predicted by your classification method.

I will compare your predicted cluster labels to the 'true' cluster labels in both `train.csv` and `test.csv` and let you know your accuracy. I'll also provide some general feedback

4.1 Functional Data Modeling

Your first goal is to explore **functional data clustering** methods such as Functional KMeans. Functional KMeans works exactly the same as finite-dimensional KMeans, but uses a functional representation of the data and function distances like L^2 to compute nearest neighbors and centroids.

1. Using the `scikit_fda` package (or analogous R pacakge), explore ways of representing the longitudinal (t_{ij}, y_{ij}) data. I recommend looking at the `BasisSmoother` object, which can be used to represent functions as a basis expansion i.e.

$$f_i(t) = \sum_n \alpha_{in}\phi_n(t) \tag{1}$$

   A good choice is the B-Spline Basis, but the package has other choices (Fourier, etc.). The `BasisSmoother` can be used to re-evaluate onto a common grid. Plot a few examples of individual subjects and their smoothed curves. Try different smoother options (basis, `smoothing_parameter`, etc.) and see how they influence the clustering result.

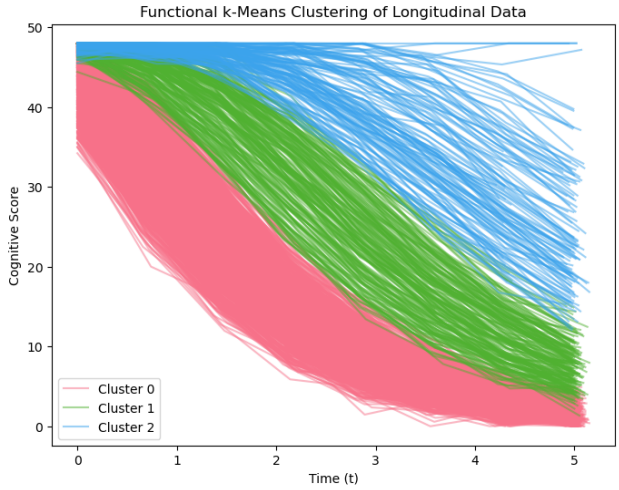

Figure 1: Color-coded plot demonstrating the cluster labels discovered by the functional KMeans algorithm. You should generate a similar plot.

2. Using either the KMeans or Agglomerative method, assign a cluster number to each subject in the training data (you can assume **three** clusters). **Note:** each clustering method expects an object of type `FDataGrid`, which you can construct using the basis smoother above. Plot the assigned clusters using color-coding like in figure 1. Append a new column called `cluster` to `train.csv` containing your results. **Note:** you will need to repeat the assigned cluster across every row of a given subject's data.

Want to try something more advanced? As an alternate to using Basis Smoothing, you can try and fit a Nonlinear Mixed Effects (NLME) model. The resulting individual-specific parameter estimates can then be used as 'low-dimensional features' to perform clustering on. The synthetic data generation method I used is essentially an NLME model run as a data generator...see Figure 2.

4.2 Classification Method to Predict Cluster from Baseline

The next goal is to see how well we can predict cluster number from measurements available at baseline.

1. Using your favorite classification method (Logistic regression, SVM, XGBoost, MLP, etc.), build a classifier that uses the features `(AGE,SEX,BIO,Y0)` to predict `cluster`, where `cluster` is the cluster number assigned by your functional data clustering method. **Note:** you can split the training data into further 'training' and 'validation' sets as part of your model building, but ultimately you will apply your method to the data found in `test.csv` and return the result to me for 'grading'

2. Apply the resulting classifier to the baseline covariates found in `test.csv` and give the result in a column `cluster`.

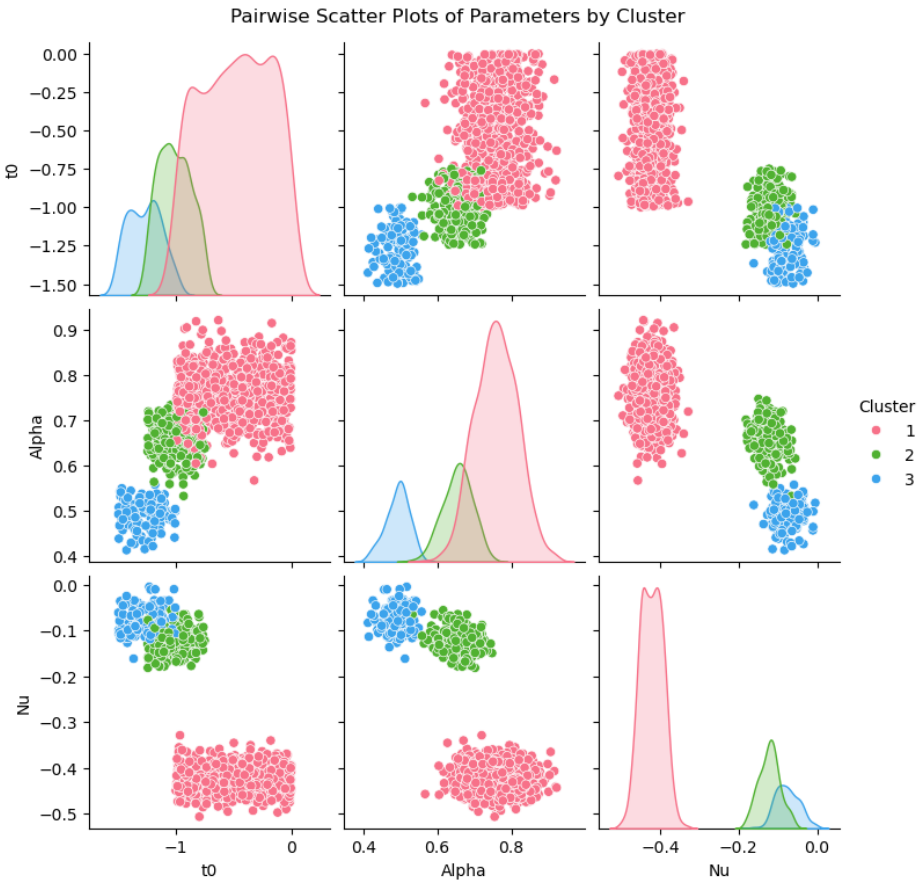

Figure 2: Scatter plot of the NLME parameters used to generate the training data, demonstrating that 'clustering is possible', at least in theory.

5 References

Here are some references if you would like to read more about ALS, functional data analysis, functional clustering, etc.

1. For a discussion on ALS clinical trial design, see Fournier, Christina N. "Considerations for amyotrophic lateral sclerosis (ALS) clinical trial design." Neurotherapeutics 19.4 (2022): 1180-1192.

2. For a general review on Functional Data Analysis see Wang, Jane-Ling, Jeng-Min Chiou, and Hans-Georg Müller. "Functional data analysis." Annual Review of Statistics and its application 3.1 (2016): 257-295.

3. For a review of functional data clustering methods see Zhang, Mimi, and Andrew Parnell. "Review of clustering methods for functional data." ACM Transactions on Knowledge Discovery from Data 17.7 (2023): 1-34.

4. For Functional KMeans theory see García, María Luz López, Ricardo García-Ródenas, and Antonia González Gómez. "K-means algorithms for functional data." Neurocomputing 151 (2015): 231-245.

5. For the 'Mixture of Gaussian Processes' method for clustering ALSFRS trajectories see Ramamoorthy, Divya, et al. "Identifying patterns of ALS progression from sparse longitudinal data." medRxiv (2021): 2021-05. and github page

6. For more background and theory on NLME models, see e.g. Davidian, Marie, and David M. Giltinan. "Nonlinear models for repeated measurement data: an overview and update." Journal of agricultural, biological, and environmental statistics 8 (2003): 387-419. These authors also have a well-regarded book on the topic.