**Guidance for using CAFCASS ECMS code**

The following code combines the Cafcass ECMS tables into the format where every row is a child on an application in a case.

*Recommended file structure*

A folder which contains all the do files, a folder for raw Stata datasets which contains three sub-folders called “Final files”, “Interim files” and “Lookups”. The do files will save all interim files (cleaned datasets) into the interim files folder, all lookups which have been imported from DB2 into the Lookups folder and the final files from the end of each do file into the Final files folder.

There are several do files and below is an explanation of what each one does. This code has been tested using the 2024/01/11 refresh from Cafcass.

**Master** – will run all of the following do files in order to create the final dataset. At the start fill in the globals with your username and password, the refresh year of the dataset you are using (note, this has been tested on the most recent Jan 2024 refresh (2024/01/11), manual checks to ensure no changes to the data are highly recommended if you change this date), and your project number within SAIL. Also, change the command directories to your own file structure.

**01create\_applications** – this cleans the applications file. It contains information by the application ID and case ID. There may be multiple applications within a case. The dataset records the application type, application date (date received, data completed etc.), court levels and court IDs. For the court levels and IDs it merges the lookup files from the CAFEREFV table. You have multiple numbers of court references: court name, court ID, region, DFJ area, circuit reference. *Note, it does not clean the lookup files themselves. If you are using for any analysis you will need to check these variables carefully because there looks like multiple instances of the courts recorded in different ways.*

**02create\_people** – this cleans the people file which contains the gender and week of birth (measured at the first Monday of the week of birth) and the person\_ID as well as the ALF IDs (for linkage to family court/census data). This is for everyone who has a person ID in the data (for anyone recorded as involved in a case). If you have requested any other person level data (which is special access), this code can be used to merge these characteristics in. The data also contains the source (e.g. self-reported, expert etc.). First, ethnicity is merged in. The code drops full duplicates, but there are still duplicates where a different ethnicity is recorded for a person. In this case the row/s in which ethnicity is unknown is dropped and the row where it is known is prioritised. Where no rows can definitively tell us the ethnicity this is recoded into unclear and all other instances dropped. *Note: this will have to be checked manually or code checked as it probably only works where there are two duplicates.* The code follows the same approach for religion, nationality, language (Cafcass say only one language is able to be recorded so any duplicates are through their own merging process). Disability it coded differently as people can have multiple disabilities. A dummy variable is created for whether they have any disability and then flags are generated for each disability they have. This code is run in separate tables for each of the characteristics and then finally merged into the main file as well as a flag for whether this information is missing. By construct an individual will have one religion, one ethnicity, one language, one nationality and multiple disabilities (flags).

**013create\_personapplication** – this cleans the persons on applications file which links the person\_id with the application\_id and identifies the role of the person on the application (e.g. subject, respondent, applicant). You should not have people with multiple roles on an application so the code looks at duplicates and removes any. Then the code merges in the people file to get the characteristics of people on applications. The sample is restricted to children (defined as anyone under the age of 18 and the subject on the application).

**014create\_linkingdatasets** – this cleans the person relationships file which details relationships between people in the dataset. Depends on the relationships of interest but this code identifies the mother and father of children on applications along with their own characteristics (merged from the people file). This is in a long format with multiple rows which means that you will have the same relationship combination but in reverse. First, the people file is merged with the related person ID to obtain the characteristics of the related person. Then, the file is reshaped so that you have one row per individual with all the related people and their characteristics in wide format. After that you can decide which relationships you are particularly interested in and run a loop to identify the related person who you are interested in. I have done this using the mother and father characteristics but you can change the relationship type to identify others e.g. grandparents/aunties. The end result is a dataset of applications relating to a person attached to the relationships and characteristics of that related person. *Note: for around 0.5% of the sample there are multiple mothers and fathers. I have not yet done anything about this, I have just taken the first instance of a “mother” for her characteristics.*

**015create\_legaloutputs** – this cleans the legal outputs file. The legal outputs are recorded per person per application only for the subject (child) on the application. It creates a flag for each legal output recorded on a file. There is a large number of these so I have sorted the legal outputs into the ones I thought were genuine legal orders and the ones that were other measures of the case and then categorised them to make the number smaller and a more manageable dataset. The names of legal outputs are cleaned and Unicode characters are removed from some of them first. If the raw data lookup file changes then this code may have to change too so may require checking with each data refresh.

**016create\_cases** – this cleans the cases file. It adds the court level IDs and the as well as the court form, law type, local authority and case type. The code adds the lookups and saves a new case file which is then merged into the main dataset.