Neuro-Integrative Connectivity (NIC) Workflow Component I: Data Processing and Transformation

Version 1.0

Section 0 — Licensing

Section 1 — Uses

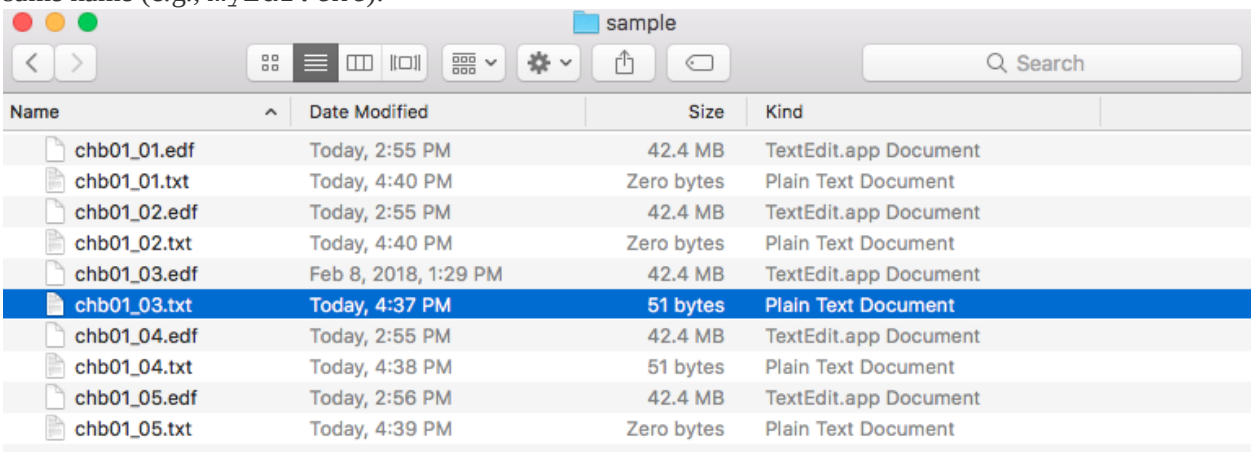
The software is intended to be used for the conversion of data files obtained from electroencephalography (EEG) in the European Data Format Plus Annotations (EDF+) into the Cloudwave Signal Format (CSF). CSF is intended to provide better human readability (based on JavaScript Object Notation), ease of computation with the companion CSF Computational Tool, and adaptability for use with parallel processing techniques.

Section 2 — Requirements

- Java SE 8 or higher is required to run the software; Java SE 10 is available at http://www.oracle.com/technetwork/java/javase/downloads/index.html.
- The software requires Javax JSON and Apache Commons Math JAR files. These should be contained in the JAR file for the converter. However, if there are issues, the required JAR files can be downloaded locally and added to the classpath during execution.
    - **JSON:** JAR files for JSON processing are included in Java EE 7 or higher; Java EE 8 is available at http://www.oracle.com/technetwork/java/javaee/downloads/index.html. Alternatively, specific JAR files may be obtained at http://download.oracle.com/otndocs/jcp/json_p-1_1-final-spec/index.html.
    - **Apache Commons:** The JAR file for Apache Commons Math is available at https://commons.apache.org/proper/commons-math/.
- EDF files should be separated into directories with one directory per patient.

Section 3 — Setup

The folder must consist of EDF files (e.g., `myEdf.edf`) with corresponding annotation text files of the same name (e.g., `myEdf.txt`).



(Note: The EDF files being used are from the CHB-MIT Scalp EEG Database, which is publicly available through the PhysioBank website.)

These text files must conform to the EDF+ specifications -- that is, they must be in a two-column format with the first column being an offset timestamp from the beginning of the file (the preferred format is HH:MM:SS.SSS), and the second column is a text annotation string. The file may indeed be empty (note that `chb01_01.txt` contains zero bytes), but it must exist.

Section 4 — Invocation

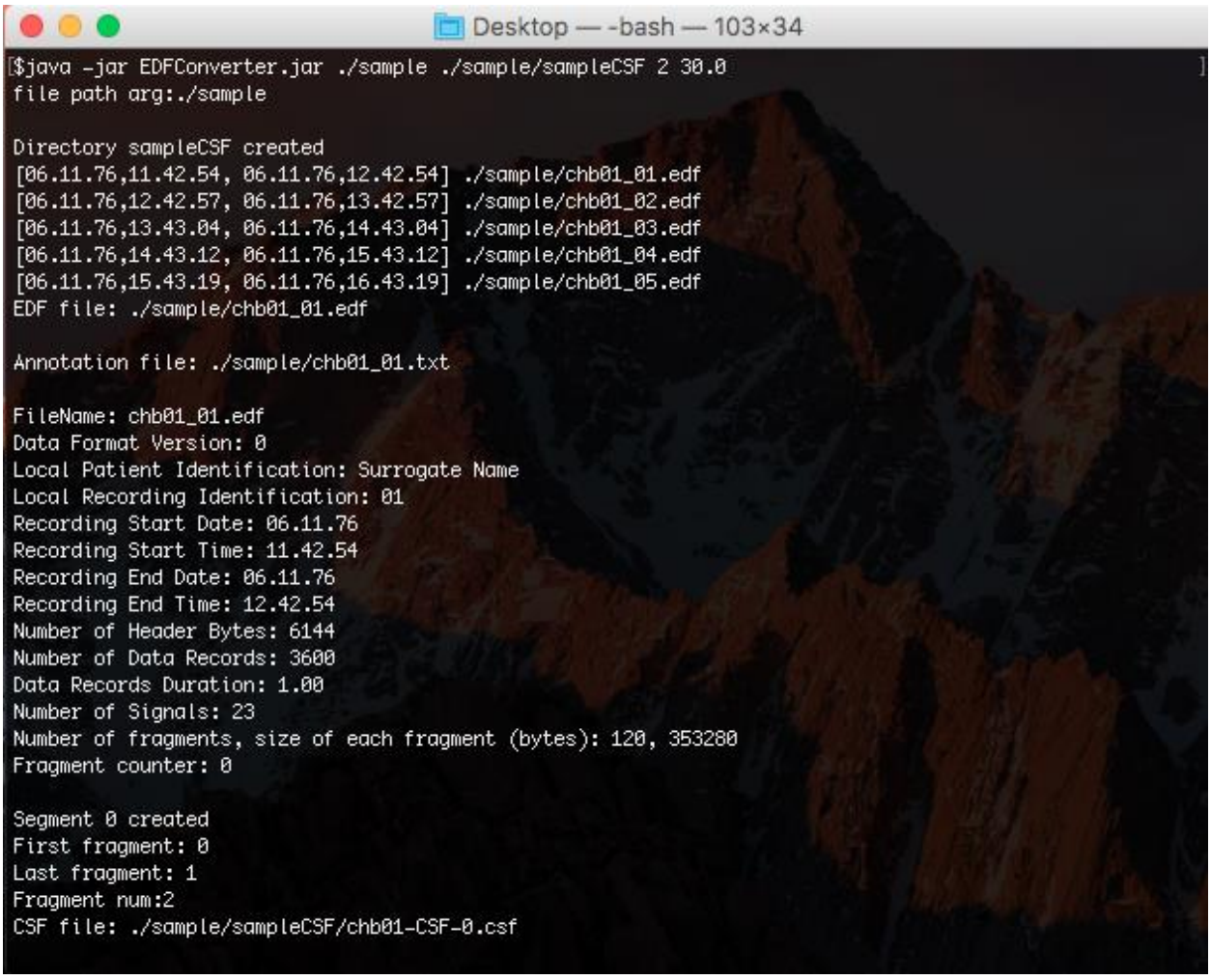
From the command line, navigate to where `EDFConverter.jar` is located and enter

```
java -jar EDFConverter.jar <EDFdir> <CSFDir> [frag {2}] [epoch {30.0}]
```

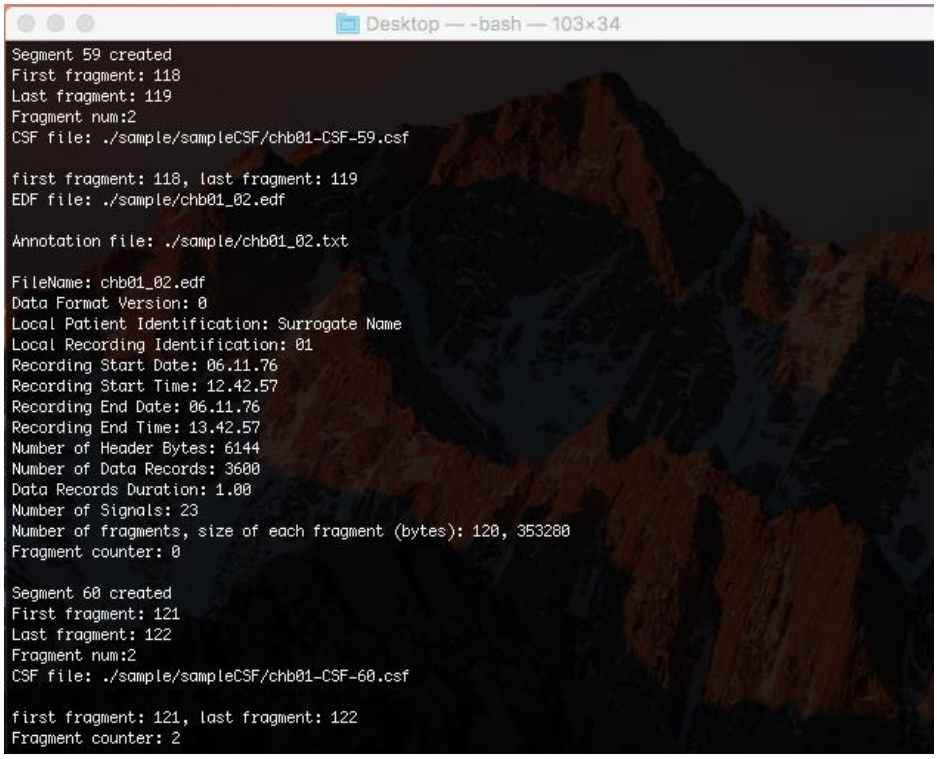
with the following arguments:

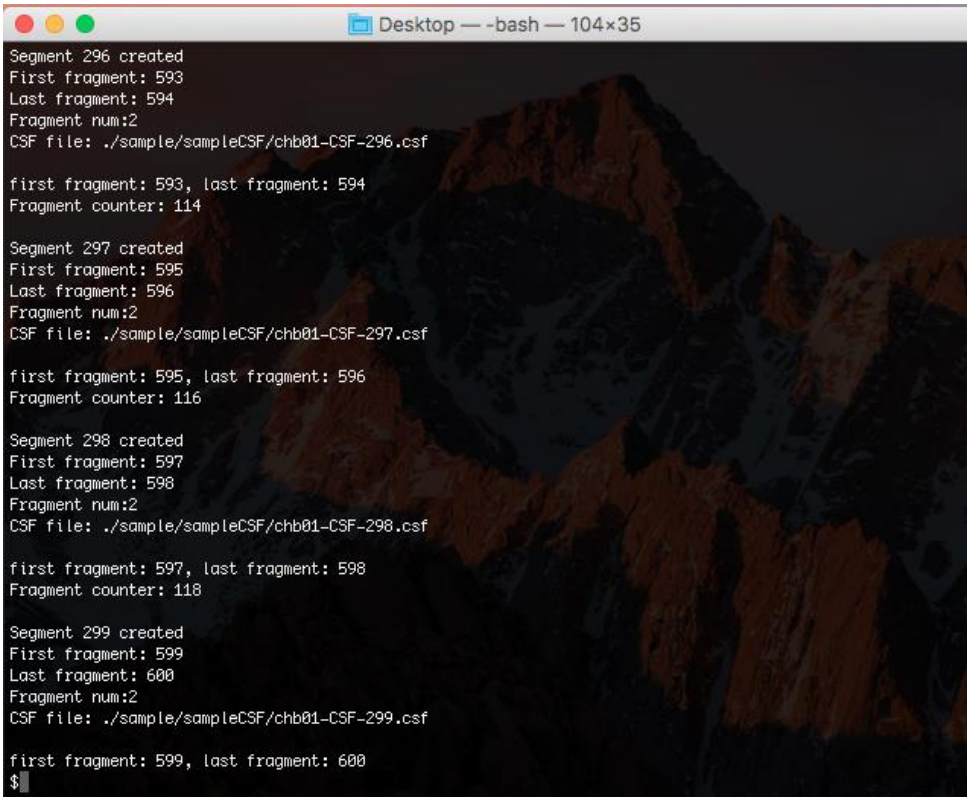
- EDFdir (mandatory): The String filepath of the directory containing the EDF files to be processed
- CSFdir (mandatory): The String filepath of the directory in which output CSF files will be placed; if the directory does not currently exist, it will be created
- Frag (optional; default 2): The integer that represents the maximum number of fragments to be included in each output CSF file
- Epoch (optional: default 30.0): The floating point value that represents the maximum time duration of each fragment in a CSF file

The program will begin by locating the destination folder for the CSF files, and creating it if it does not already exist. It will then sort the EDF files into chronological order. Then the program will process the EDF files in order, obtaining the EDF file and the associated annotation text file, and thereby create the corresponding CSF files. The program prints header information for each EDF file (this is for verification purposes) and summary statistics about the creation of each CSF file.



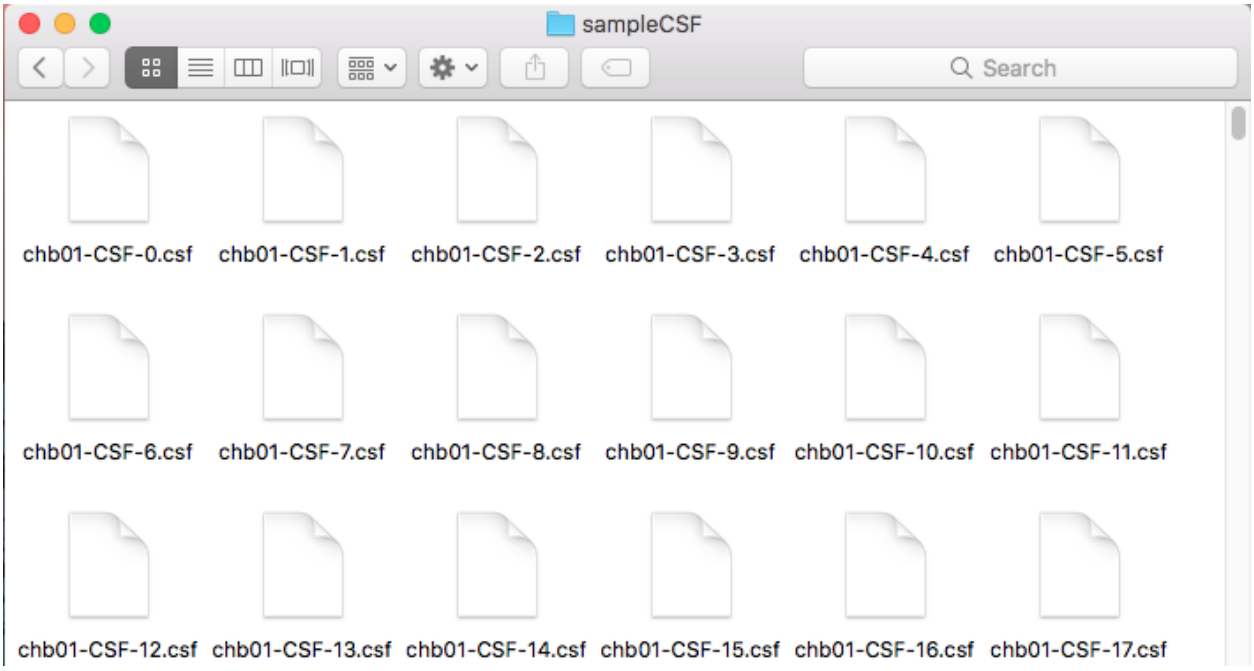
The program will continue in this manner until all of the EDF files have been processed.





Section 4 — Output

The output will be a directory populated with CSF files resulting from the processing of the directory of EDF files. The names of the CSF files are of the form `myEDF-CSF-#.csf`, where `myEDF` is the name common to the EDF files that occurs before the first underscore character, and the numbering of the files is chronological beginning at 0 (the EDF files are sorted chronologically before processing). Information about the correspondence between CSF files and EDF files is contained within the CSF files themselves, and is therefore not reflected in the filenames.



Credits

This software has been developed at Case Western Reserve University as part of a research project and includes contributions by the NIC development team members: Arthur Gershon, Pramith Devulapalli, Vimig Socrates, Haroon Khazi, and Satya Sahoo (Project PI).