**Dokumentacja programu – miony**

1. Jak korzystać z programu?

Należy w katalogu **input/** umieścić interesujące nas pliki mionów, a następnie uruchomić program. Program wygeneruje pliki gotowe do Matlaba w folderze **matlab/**

Pliki BEFORE zapisywane są w folderze **matlab/CHANNEL/before**, a pliki AFTER analogicznie w katalogu **matlab/CHANNEL/after** (gdzie CHANNEL odpowiada kanałom: ch0, ch1, ch2).

Uwaga! Program automatycznie tworzy poniższe foldery:

* matlab/
* matlab/CHANNEL/before
* matlab/CHANNEL/after
* input/
* binary/
* bessel/
* dampings/
* dampings/angles
* dampings/init\_point
* dampings/energy
* histograms/

Po pierwszym uruchomieniu stworzy się folder **input/** do którego można umieścić pliki tekstowe do konwersji w formacie:

TIME BEFORE AFTER (3 kolumny)

Pierwsze uruchomienie wywołuje również utworzenie folderu **bessel/.**

Jeśli folderu nie ma, można go stworzyć samemu. To samo w przypadku folderu **binary/**, tworzy się on gdy użytkownik programu wybierzę opcje konwersji z plików tekstowych na pliki binarne.

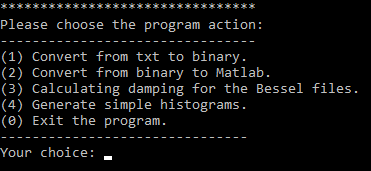
Katalogi **matlab/**, **binary/**, **dampings/, histograms/** czyszczą się przy każdym wykonaniu odpowiednich dla nich operacji. Oznacza to, że gdy konwertujemy pliki tekstowe z katalogu input/, w binary/ uzyskamy tylko pliki binarne z tego katalogu (dlatego wykonywane jest czyszczenie). Tak samo w pozostałych sytuacjach.

Katalog bessel/ służy do liczenia tłumienia. Do tego katalogu umieszczamy pliki przekonwertowane w Matlabie. Następnie odpowiednia opcja w programie policzy nam tłumienie i zapisze je do pliku dampings.txt. Gdy chcemy policzyć tłumienie dla 3 kanałów w oddzielnych plikach, najlepiej na początku umieścić pliki dla jednego kanału, uruchomić program z opcją liczenia tłumienia, usunąć pliki z katalogu bessel/ i ponownie wgrać pliki z innego kanału.

W katalogu dampings/ generowane są pliki tłumienia dla poszczególnych kategorii. Pliki znajdziemy w katalogach dampings/angle, dampings/energy, dampings/init\_point. Po przekonwertowaniu plików bessels na kąty, energie i init\_point znajdują się w tych katalogach odpowiednie pliki.

Histogramy generowane są do katalogu histograms/, gdzie są podzielone na dwa pliku dla kanałów 6\_60MHz i 2\_80MHz.

1. Opcje programu



Po uruchomieniu programu pojawi się konsola z panelem wyboru akcji. Po kolei opiszemy działanie każdej z nich.

1. Konwertuje pliki z katalogu input/ z tekstowych na binarne do katalogu binary/
2. Konwertuje pliki z katalogu binary/ na Matlabowe w katalogach matlab/CHANNEL/before i matlab/CHANNEL/after.
3. Liczy tłumienie z plików w katalogu bessel/ do pliku dampings.txt w głównym katalogu.
4. Generuje histogram z plików tłumienia. Po uruchomieniu należy podać nazwę pliku z którego generowany ma zostać histogram. W przypadku gdy plik znajduje się w katalogu programu, należy podać ścieżkę katalog\kategoria\plik.txt (przykład: **dampings\angle\70**).

Ostatnia opcja służy do wyjścia z programu.

Odpowiednią opcje wpisujemy (numerek) w konsoli, a następnie uruchomi się żądana akcja.

1. Opis struktury programu

**Klasy / struktury POJO:**

* Bessel – jest strukturą, do której czytane są dane z plików Bessel’a.
  + double before – zawiera informacje o kanale before z pliku Bessela.
  + double six60MHz – zawiera informacje o kanale 6\_60MHz z pliku Bessela.
  + double two80MHz – zawiera informacje o kanale 2\_80MHz z pliku Bessela.
* Damping – jest strukturą, do której zapisywane są tłumienia w formacie:
  + double six60Mhz – tłumienie dla kanalu 6\_60MHz.
  + double two80MHz – tłumienie dla kanału 2\_80MHz.
* SingleBlock – jest strukturą, która przechowuje jeden wiersz z pliku wejściowego input o formacie:
  + long time – zawiera informacje z pierwszej kolumny plików wejściowych z symulacji OFFLINE.
  + long before – zawiera informacje o drugiej kolumnie plików wejściowych symulacji OFFLINE, czyli informacje przed filtrowaniem.
  + long after – zawiera informacje o trzeciej kolumnie plików wejściowych symulacji OFFLINE, czyli informacja po filtrowaniu.
* BesselName – jest strukturą, którzy przyjmuje dwie wartości: stara nazwa pliku i nowa nazwa pliku na podstawie których jest w stanie przyporządkować nazwy do plików Bessel’a typu:
  + 0000 0000\_68\_70\_1E7\_1000\_0\_i-017cabe1cb877509d\_r1\_MER.txt.ch1\_org

Na podstawie tej struktury dowiązywane są później pliki Bessel przy tworzeniu kategorii angle, init point i energy.

**Klasy główne:**

* Converter – klasa odpowiada za każdą konwersję, która następuje po wybraniu odpowiedniej akcji w programie. Klasa posiada poniższe metody:
  + separateAndSaveChannels(pathToFiles, binaryFolder) – klasa separuje dane po kanałach z danego pliku wejściowego, a następnie zapisuje je do pliku binarnego (z każdego pliku tekstowego powstają 3 pliki binarne \_ch0, \_ch1, \_ch2).
  + convertBinaryToMatlab(pathToFiles, matlabFolder) – klasa konweruje pliki binarne na format Matlaba, dzięki czemu możemy je potem bez problemu odczytać w programie. Każdy kanał jest zapisywane oddzielnie. Następnie jest podzielony na dwie części before i after. Powyżej w dokumentacji opisane są foldery.
  + convertBesselToDamping(pathToFiles) – metoda konwertuje pliki po filtrze Bessel’a na pliki wyjściowe z tłumieniem.
  + generateHistogram(…) – metoda generuje histogram dla podanego pliku wejściowego z tłumieniem, oddzielne dla obu kanałów.
* Loader - klasa służy do wczytywania podstawowych struktur SingleBlock i Bessel.
  + getBlocksFromBinaryFile(filename) – podany filename oznacza nazwę pliku binarnego lub nazwę z jego lokalizacją np (binary\\test.bin), po czym metoda wczytuje plik binarny do vectora struktur SingleBlock. Zwraca vector<SingleBlock> z danymi.
  + getBesselFromFile(filename) – wczytuje z podanego pliku lub nazwy z lokalizacją do vectora ze strukturą Bessel dane z całego pliku. Zwraca vector<Bessel> z danymi.
* Histogram – klasa służy do generowania histogramu, zawiera metodę:
  + generateSimpleHistograms() – metoda generuje histogramy dla dwóch kanałów 6\_60MHz i 2\_80MHz w oddzielnych plikach. W metodzie generowany jest cały wygląd histogramu.
  + generateLabels() : private – generuje labelki dla histogramu (metoda prywatna, używana wewnątrz generateSimpleHistograms)
  + Konstruktor Histogram(...) – przyjmuje vector<Dampings> czyli listę tłumień z których mają być tworzone histogramy, zakres od – do oraz liczbę przedziałów. Niestety nie ma uniwersalnego rozwiązania, dlatego histogram ma domyślnie dwa zakresy z których można generować histogram i użytkownik może zdecydować który zakres chce wybrać w interfejsie.
  + generateCsv() : private – metoda prywatna do generowania pliku w formacie CSV z danymi z histogramu. Dzięki tej metodzie możemy w łatwy sposób wykonać import danych do Excela, a następnie stworzyć do nich wykresy.

**Klasy pełniące funkcje pomocnicze:**

* BesselHelper – klasa pomocnicza dla stuktur Bessel’a. Klasa zawiera metody statyczne:
  + getMaximumBessel(bessels) – przyjmuje vector Bessel, a następnie zwraca strukture Bessel z danymi o maksymalnej wartości dla before, six60MHz i two80MHz.
  + calculateDampingFromBessel(bessel) – przyjmuje pojedynczą strukturę Bessel, z której liczy tłumienie i zwraca strukturę Damping z gotowym tłumieniem dla danej struktury.
  + getDampingsFromBesseels(bessels) – metoda przyjmuje vector bessels, w którym szuka potrójnych maximów i liczy dla nich tłumienie, a następnie zwraca vector dampings z policzonymi tłumieniami dla podanego vectora bessels.
  + getMaximumFromCompartment(indexes, maximums, compartment) – metoda pomocnicza, służy do wyszukania maximum dla przesłanego kanału np. 2\_80MHz, używając tablicy indeksów z before. Metoda usuwa maximum z listy, jeśli spełnia ono warunki i jest ono znalezione w określonym przedziale. Metoda przyjmuje rozpatrywany przedział, dla którego szukane są maxima.
  + getMaximumFromMap(values) – metoda służy do znalezienie maximum z podanej mapy formatu <int, double>, poszukując najwyższej wartości double. Zwraca parę <index, wartość>, gdyż index jest później potrzebny do operacji.
  + getNumberOfCompartment(indexes, numOfIndex) – dla podanego indexu (jest to liczba pomocnicza, która określa numer, na którym występuje dana wartość) szuka, czy liczba została złapana w jakiś przedział. Jeśli tak to zwraca numer przedziału. Jeśli nie to zwraca -1.
* FilesHelper – klasa pomocnicza dla pracy z plikami. Klasa zawiera metody statyczne:
  + getPathToFilesFromDirectory(dirName) – klasa przyjmuje nazwę katalogu, z której zwraca vector ze wszystkimi ścieżkami do plików znajdujących się w podanym katalogu.
  + splitStringToDouble(...) – przyjmuje string, delimiter oraz referencje do vectora double. Metoda realizuje konwersję stringa, który ma dane oddzielone przez określony delimiter do vectora double.
  + splitFilenameByDelimiter(...) – przyjmuje string, delimiter oraz referencje do vecotra stringów. Metoda realizuje konwersję nazwy pliku oddzielonego delimiterem na poszczególne składowe. Dzięki niej możemy rozdzielić np 00\_test\_plik na vector zawierający trzy elementy 00, test, plik.
  + saveDampingToFile(damping, filename) – metoda przyjmuje tłumienie i nazwę pliku, do którego zapisze dane.
  + saveSingleBlocksToBinary(singleBlocks, filename) – metoda zapisuję vector SingleBlock do pliku binarnego.
  + getFilenameForDamping(filename) – metoda przyjmuje nazwę pliku wejściowego Bessel, po której rozpoznaje do którego pliku zapisać dane wyjściowe (w zależności od kanału ch1, ch2 lub ch2).
* HistogramHelper – klasa realizująca pomocne operacje dla histogramu. Klasa zawiera metodę statyczną:
  + calculatePercentage(counted, totalSize) – metoda przyjmuje policzone ilości wystąpień danych w postaci vectora zawierającego przedziału, z czego wylicza wartości procentowe i zwraca vector z procentami.
* BesselNameHelper – klasa realizuje pomocne operacje dla powiązań nazw plików Besselowych. Posiada następującą metodę statyczną:
  + getBesselNameForSelectedFile(besselsNames, filename) – metoda przyjmuje vector struktury BesselName z powiązaniami dla nazw plików oraz nazwę pliku, dla którego chcemy uzyskać odpowiadający rekord BesselName który będzie powiązaniem. Metoda zwraca pojedynczą strukturę BesselName, która zawiera starą i nową nazwę pliku.

**Klasa interfejsu:**

* UserInterface – klasa odpowiada za wyświetlenie w konsoli interfejsu wyboru dla użytkownika programu, dzięki czemu umożliwia interakcję i różne sposoby działania programu. Klasa zawiera metody:
  + getUserChoice() – metoda pobiera od użytkownika wybór opcji, a następnie zwraca int w zależności od opcji którą wybrał.
  + runUserAction() – główna metoda, która zarządza całym interfejsem, używa metody getUserChoice(), na podstawie której wybiera odpowiedni zestaw instrukcji.