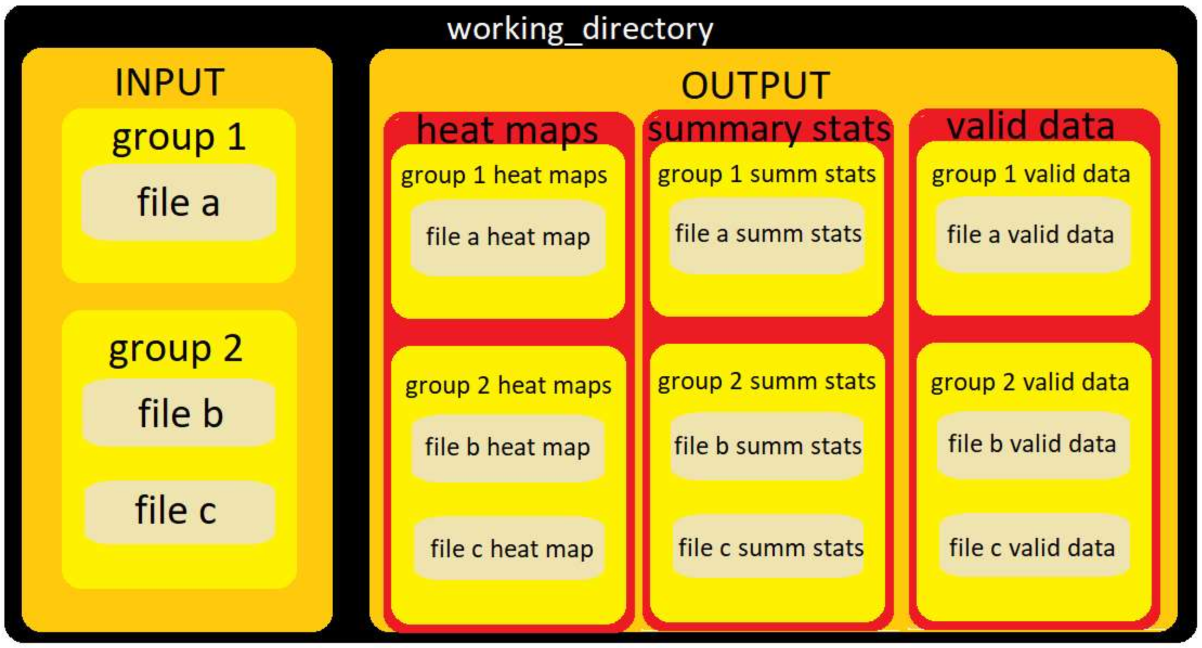
Jakob Arend

January 14, 2022

activPAL SLNW algorithm and data reduction guide and documentation

GENERAL GUIDE

This guide will focus on the high-level representation of the data reduction. For specifics on each function, please see the documentation below. To run this code, you will need to format your input in a very specific way. First, choose a working directory (this can be any folder). Next, create a folder named "INPUT" in your chosen directory. Inside of this folder, deposit all your activPAL data. This data should consist of any number of groups represented as folders, which contain that groups respective .csv files. Every .csv file needs to be within a group folder—even if there is only one .csv inside of that group folder. If you are having trouble understand the input setup, refer to the graphic below. Once you have your input configured properly, navigate to line 478 (3 lines from the bottom). This line contains the variable working_directory—change this variable to contain the path to the working directory you chose earlier. To run the code, simply run all lines from the beginning. Note that this code clears the environment before and after use with the first and last lines of code. If you would like to preserve a previous environment or keep some of the environment loaded, comment out the respective lines. Once run, this code will generate heat maps, summary statistics, and valid data for the .csv files contained in "INPUT". This output will map to a new folder called "OUTPUT". You do not need to create this folder; the code will create every output folder it needs. Note here that if you call this code in a directory that already contains a folder named "OUTPUT", it will add to that folder AND POSSIBLY OVERWRITE IT. Take caution to ensure that your working directory does not have a folder named "OUTPUT" in it when you run this code.



DETAILED FUNCTION DOCUMENTATION

main(working_directory)

parent function: N/A
input:

Character string "working_directory" which contains the path to the working directory.
implicit input: INPUT folder exists in directory—more on this in assumptions

output:

Creates folder "OUTPUT" in the current working directory. Creates folders "heat_maps", "summary_statistics", and "valid_data" inside of folder "OUTPUT". Creates separate folders for each group present in the INPUT folder within each of these three subfolders. Populates these folders with appropriate heat maps, summary statistics, and valid data from post-algorithm data

Example: If there are two group folders in INPUT titled "A" and "B", main will create a folder OUTPUT that contains three folders heat_maps, summary_statistics, and valid_data. Each of these three folders will contain two folders titled for example "A heat maps" and "B heat maps". These two folders will contain the heat maps from the activPAL SLNW algorithm when run on the .csv files, and likewise for the summary statistics and valid data.

description:

Main is a function wrapped around the entirety of the code for ease of use. Input your working directory and main will run the activPAL SLNW algorithm, generating activity heat maps, summary statistics, and returning the valid data left over.

assumptions:

- assumes that there exists a folder within working_directory named "INPUT" which contains groups of activPAL .csv files contained in separate folders inside "INPUT".

Note: main contains many more assumptions, but those assumptions are dealt with in its helper functions. Thus, to make bug finding easier I will address those assumptions when they arise in their specific helper functions.

create_directories(working_directory)

parent function: main
input:

Character string "working_directory" which contains the path to the working directory.

output:

Creates folder "OUTPUT" in the current working directory. Creates folders "heat_maps", "summary_statistics", and "valid_data" inside of folder "OUTPUT". Creates separate folders for each group present in the INPUT folder within each of these three subfolders.

description:

Create_directories creates the directories necessary for later output heat maps, summary statistics, and valid data to be deposited in.

assumptions:

- assumes that "working_directory" is a valid directory

SLNW(file)

parent function: main
input:

Character string “file” which contains the path to the activPAL .csv file to run the activPAL SLNW algorithm on.

output:

Creates a heat map of post-algorithm valid activity data and prints it to the heat_maps folder. Returns post-algorithm valid activity data in a .csv.

description:

SLNW runs the activPAL SLNW algorithm on an input .csv file and outputs a heat map and valid activity data. This function is a helper function to main and is called repeatedly by it on each file in INPUT to generate all heat maps and valid data .csv’s.

assumptions:

- assumes that file is a valid path
- assumes that the .csv file obtained from that path is an activPAL .csv
- assumes that the data in this activPAL .csv is formatted properly—this means that function activpal.file.reader() from package activpalProcessing can be run on the input file without error or unreasonable output, check activpalProcessing documentation here for specifics

get_summary_statistics (file, data, instructions)

parent function: main
input:

Character string “file” which contains the path to output calculated summary statistics to.
Dataframe data which contains the post-algorithm valid data to get summary statistics from.
Dataframe instructions which contains the specific groups to calculate summary statistics for.

output:

Creates a .csv of summary statistics (count, average, minimum, maximum, median, interquartile range, standard deviation, and coefficient of variation) on data based on instructions (for details on what the dataframe instructions should look like see function format_instructions).

description:

Get_summary_statistics takes in post-algorithm valid data and calls pnc_summary_statistics repeatedly to calculate summary statistics on the bouts in data for each group in instructions, by each day in valid data and for all days combined.

*Example: If get_summary_statistics is called with 3 groups in instructions and there are 5 different days within the dataframe data, then there will be 3*5=15 different columns plus 3 more columns for all days combined for a total of 18 columns. Each of these columns has values for 8 different statistics listed above.*

assumptions:

- assumes that file is a valid path to output to
- assumes that data is post-algorithm valid data (proper columns and data types, no breaks, NaN’s, Inf’s, etc.)
- assumes that instructions is a properly formatted dataframe in compliance with the method laid out in function format_instructions

format_instructions(instructions)

parent function: main, get_summary_statistics

input:

List “instructions” which contains a continual sequence of separate instructions. Each instruction is formatted as “name, type, lower bound, upper bound, lower exclusive, upper exclusive”. Name is a character string that will be the name output in columns in summary statistics (for example if the instruction describes sedentary bouts less than 5 minutes try “sed <5” for the name). Type is a character string that refers to the activity type, with “Sedentary”, “Stepping”, and “All” to represent activities lying/sitting, stepping, and lying/sitting/stepping respectively. Note that the type MUST be one of these three strings. Lower bound is a numeric which refers to the lower bound of the instruction, in the earlier example of sed <5 lower bound would be 0. Upper bound is a numeric which refers to the upper bound of the instruction and for sed <5 would be 5*60=300. Note here that bounds are assumed to be in seconds. Lower exclusive is a logical which indicates the lower bound is exclusive if TRUE and inclusive if FALSE. Upper exclusive is a logical which indicates the same for the upper bound. These instructions are arranged one after the other with no breaks or compartmentalization.

*Example: An example set of instructions for sedentary bouts less than 20 mins (exclusive), more than 20 mins (inclusive), and then all sedentary bouts would be list("sedentary less than 20 mins", "Sedentary", 0, 20 * 60, TRUE, TRUE, "sedentary more than 20 mins", "Sedentary", 20*60, Inf, FALSE, TRUE, "sedentary all bouts", "Sedentary", 0, Inf, TRUE, TRUE).*

output:

Rearranges this list into a dataframe with appropriate headers for get_summary_statistics to use.

description:

Format_instructions takes in a list of consecutive instructions and reshapes it into a dataframe with appropriate headers for get_summary_statistics. This function is primarily for ease of use.

assumptions:

- assumes that the input list instructions is formatted as described above in section “input” of this function

pnc_summary_statistics (data)

parent function: get_summary_statistics

input:

Numeric vector data

output:

A numeric vector containing the count, average, minimum, maximum, median, interquartile range, standard deviation, and coefficient of variation of the input data

description:

Pnc_summary_statistics calculates summary statistics for an input numeric vector. This function is called repeatedly by get_summary_statistics on vectors containing the lengths of bouts filtered by each specific instruction.

assumptions:

- assumes that the input numeric vector contains all real, finite numerics