activAnalyzer: An R Shiny app to analyse ActiGraph accelerometer data and to implement the use of the PROactive Physical Activity in COPD instruments

Pierre-Yves de Müllenheim and Arnaud Chambellan

1 Institut de formation en éducation physique et en sport d’Angers (IFEPSA-UCO), Les Ponts-de-Cé, France
2 Groupement des Hôpitaux de l’Institut Catholique de Lille (GHICL), Hôpital Saint Philibert, Lomme, France

Statement of need

ActiGraph devices (ActiGraph LLC, Pensacola, FL) have been the most used accelerometers in scientific literature to measure physical activity (Bassett et al., 2015; Migueles et al., 2017). These devices, along with their software companion ActiLife (ActiGraph LLC, Pensacola, FL), allow to get movement data expressed in either activity counts or G-force units. Activity counts represent the amount of acceleration produced over a given epoch of time at the wearing position of the device. While analytic methods based on G-force data are developing, using activity counts to assess physical activity and sedentary behaviour remains common (Migueles et al., 2017). Interestingly, ActiGraph devices have been validated to implement the framework related to the use of the PROactive Physical Activity in COPD (chronic obstructive pulmonary disease) instruments (Dobbels et al., 2014; Garcia-Aymerich et al., 2021; Gimeno-Santos et al., 2015). Such a framework requires combining scores relating to answers of questionnaire items and scores relating to accelerometer metrics obtained from a week of measurement.

Due to the large size of data files to analyse when using accelerometer and due to the relative complexity of the implementation of some algorithms, a simple spreadsheet does not appear to be a feasible tool to complete all the accelerometer data analysis workflow from the same place. The Full version of ActiLife software allows completing all the general steps of the data analysis workflow with activity counts but the cost of this ActiLife version may prevent teaching a wide audience to implement such a workflow and may also prevent working with large teams on recorded data (Lite versions of ActiLife software, that are at a lower cost than Full versions, allow device initialisation and data downloading only). Moreover, there is no solution in ActiLife software to fully implement the PROactive framework for COPD patients described above.

Other ways than ActiLife software to analyse activity counts include using programming languages. R (R Core Team, 2022) and Python (Python Software Foundation, 2022) have been programming languages commonly used by scientists to build tools aiming at fostering physical activity data analysis. In R, the accelerometry and nhanesaccel packages by Van Domelen and Pittard (2014), the actigraph.sleepr package by Petkova (2021), and the pawacc package by Geraci (2017), provide several functions to perform analyses of interest with activity counts. In Python, the pyActigraphy library by Hammad and Reyt (2020) also allows, among various other features, to handle ActiGraph activity counts. While useful for research settings, these resources may be of a little interest for other settings where people have no programming skills, because they do not propose a GUI (graphical user interface) to help people who do not code and who have no time to learn this skill. Beyond the lack of a free and simple interface to analyse ActiGraph activity counts data, there is, to our knowledge, no app that allows an easy implementation of the PROactive framework with...
COPD patients that would be based on an analysis of ActiGraph activity counts. This is why we have developed the activAnalyzer app. For now, a first main interest of this app is to allow teaching large groups of students or professionals, who have no programming skills, to analyse activity counts for assessing physical behaviour. A second main interest is to allow an easy implementation of the PROactive framework with COPD patients when working with an ActiGraph accelerometer, this by clinicians, healthcare providers and/or researchers, either in clinical routine or in research setting.

Functionality and purpose of the software

activAnalyzer is an app built as a package using R programming language. The app can be used in three different ways as explained elsewhere (https://pydemull.github.io/activAnalyzer/), including (i) a standalone desktop application for Windows machines only thanks to the DesktopDeployR framework developed by Lee Pang; (ii) using R and RStudio software along with the CRAN version of the activAnalyzer package or its development version from GitHub; (iii) using a shinyapps.io platform. If used with R and RStudio, the app will require the installation of the TinyTeX distribution to generate .pdf reports, as explained on the app website.

When the user opens the app, they has to deal with four ordered sections. The first section allows the user to complete information related to the measurement setup (patient’s characteristics, device position, etc.). In the second section, the user must upload an .agd data file (’.agd’ being the extension of the initial file generated by ActiLife software when the user wants to work with activity counts data). Then, the user has to configure the app to detect nonwear time. The results from this first analysis can be visualised by the user, as shown in Figure 1.

In a third section, the user has to select an equation to estimate energy expenditure and values to define cut-points in counts/min. Cut-points are the values below or above which one can be classified as being in sedentary behaviour or in light, moderate, or vigorous physical activity. Once the user has completed the configuration for intensity analysis, they can run analysis. Then, the user can see a figure showing time spent in the different categories of activity intensity (Figure 2), a table showing the results of the measurement for each day (Figure 3), and tables with daily means and daily medians, respectively, showing metrics summarised from valid days (e.g., Figure 4 for means).

Figure 1: Example of analysis for nonwear time detection.

Figure 2: Example of analysis for activity intensity categories.

Figure 3: Example of analysis for daily means and medians.

Figure 4: Example of analysis for valid days.

Once analysis is finished, the user can generate a report of the measurement, download .csv files containing data produced by the app, or go to the questionnaires related to the PROactive framework. This last part consists of completing the chosen questionnaire, and downloading a report once analysis is completed.

Acknowledgements

The authors thank Florian Congnard and Bénédicte Noury-Desveaux for their valuable feedback. The authors also thank Sebastien Chastin and Ulf Ekelund for giving time to provide their point of view on some of the figures the app can generate for the reports and that use data they published. Of note, the final decision regarding the use of the concerned figures and their design rested with the authors of the present work.
References


