Flexible R Functions for Processing Accelerometer Data, with Emphasis on NHANES 2003–2006

by Dane R. Van Domelen and W. Stephen Pittard

Abstract Accelerometers are a valuable tool for measuring physical activity (PA) in epidemiological studies. However, considerable processing is needed to convert time-series accelerometer data into meaningful variables for statistical analysis. This article describes two recently developed R packages for processing accelerometer data. The package **accelerometry** contains functions for performing various data processing procedures, such as identifying periods of non-wear time and bouts of activity. The functions are flexible, computationally efficient, and compatible with uniaxial or triaxial data. The package **nhanesaccel** is specifically for processing data from the National Health and Nutrition Examination Survey (NHANES), years 2003–2006. Its primary function generates measures of PA volume, intensity, frequency, and patterns according to user-specified data processing methods. This function can process the NHANES 2003–2006 dataset in under one minute, which is a drastic improvement over existing software. This article highlights important features of packages **accelerometry** and **nhanesaccel** and demonstrates typical usage for PA researchers.

Introduction

The National Health and Nutrition Examination Survey (NHANES) is one of a select few national studies to include objective physical activity (PA) monitoring with accelerometers (Troiano et al., 2008; Hagstromer et al., 2007; Colley et al., 2011). In the 2003–2004 and 2005–2006 waves of the study, a total of 14,631 Americans age ≥ 6 years wore uniaxial ActiGraph GT1M accelerometers at the hip for seven consecutive days. The result is a rich dataset for describing PA levels in Americans and assessing correlates of PA.

In 2007, SAS programs for processing the accelerometer data from NHANES were made available on the National Cancer Institute's (NCI) website (National Cancer Institute, 2007). Using these programs, researchers around the world could access the time-series accelerometer data from NHANES and run SAS programs to derive meaningful PA variables to use in statistical analyses. Over the past seven years, the NCI's SAS programs have facilitated great progress in understanding the distribution of PA behaviors in America (Troiano et al., 2008; Tudor-Locke et al., 2010), and have helped to identify numerous cross-sectional associations between PA and health outcomes (Healy et al., 2011; Sisson et al., 2010; Carson and Janssen, 2011).

While the NCI's SAS programs provide several indicators of overall PA and moderate-to-vigorous PA (MVPA), researchers are increasingly writing customized scripts in programs such as MATLAB (The MathWorks, Inc., 2014), LabVIEW (National Instruments, 2014), and R for more flexibility in data processing (Tudor-Locke et al., 2011). This approach allows researchers to generate more descriptive variables, such as time spent in extended periods of sedentariness or hour-by-hour count averages (Bankoski et al., 2011); to implement non-wear algorithms that might better discriminate between true non-wear and sedentary time (Choi et al., 2012); and to apply activity intensity cut-points specific to certain populations, such as children and older adults (Copeland and Esliger, 2009).

As the level of complexity in data processing methods increases, the learning curve for researchers interested in working with accelerometer data also increases. More generally, researchers with data from studies other than NHANES are tasked with writing their own software to convert the time-series data to a form suitable for statistical analysis. Such efforts are time-consuming and probably inefficient, as many different researchers are likely to write their own code for very similar purposes. The size of accelerometer files causes additional complexity. For example, the accelerometer files in NHANES 2003–2006 are 3.87 GB and contain over 1.25 billion data points.

The purpose of this paper is to describe recently developed software for processing accelerometer data. The software takes the form of two R packages: **accelerometry** for general functions (Van Domelen, 2014), and **nhanesaccel** for processing NHANES 2003–2006 data (Van Domelen et al., 2014).

Table 1: Design goals for R package **accelerometry**.

Goal	Description
1	Allow flexibility in data processing (non-wear algorithm, bouts, etc.).
2	Able to process data from uniaxial or triaxial accelerometers worn at the hip, wrist, or other position.
3	Run efficiently so that large datasets can be processed quickly.
4	Generate a wide variety of physical activity variables, including those that can be used to study weekday/weekend and time-of-day effects.
5	Use reasonable defaults for function inputs so that users with minimal knowledge of accelerometry can still process their data adequately.
6	Free.

R package 'accelerometry'

Overview

The package **accelerometry** has a collection of functions for performing various tasks in accelerometer data processing, as well as composite functions for converting minute-to-minute uniaxial or triaxial accelerometer data into meaningful PA variables. This package is intended for researchers who wish to process accelerometer data from studies other than NHANES.

Design goals

The package was developed according to the design goals in Table 1. Computing speed was considered particularly important because users may have data on thousands or tens of thousands of participants. Such a large volume of data can take hours to process using inefficient software.

Implementation

Using R with embedded C++ code was a natural choice to meet goals 3 and 6, and the functions in the package were written to achieve goals 1–2 and 4–5. The package contains eleven functions for performing various steps in accelerometer data processing. These functions are summarized in Table 2. For seven of the functions, C++ code was added via the **Rcpp** package (Eddelbuettel and François, 2011; Eddelbuettel, 2013) to improve efficiency.

Most users will only interact with `accel.process.uni` or `accel.process.tri`. These functions internally call the other functions, and have inputs to control the non-wear and activity bout algorithms, count cut-points for activity intensities, and other options. Notably, `accel.process.tri` allows the user to choose which accelerometer axis to use for non-wear detection, activity bouts, and intensity classification. This is important because vertical-axis counts are almost always used for intensity classification, but triaxial data is more sensitive to small movements and therefore well-suited for non-wear detection (Choi et al., 2012).

Version 2.2.4 of **accelerometry** is currently available on CRAN. The package requires R version 3.0.0 or above due to its dependency on **Rcpp**, which itself requires R 3.0.0 or above. A data dictionary for the variables generated by the functions in **accelerometry** is available online (Van Domelen, 2013).

Examples

Suppose a researcher wishes to process triaxial data collected from an ActiGraph GT3X+ device (ActiGraph, Pensacola, FL) worn at the hip for a particular study participant. The first step is to load a data file with the accelerometer count values in one-minute intervals ("epochs") into R. After transferring data from the accelerometer to a computer via USB cable, the researcher can use Actigraph's ActiLife software (ActiGraph, 2014) to create a .csv file and load it into R using `read.csv`, or use the function `gt3xAccFile` in the R package **pawacc** (Geraci, 2014; Geraci et al., 2012) to load the accelerometer file directly into R. Notably, data collected in shorter than one-minute epochs can still be processed using **accelerometry**, it just has to be converted to one-minute epochs first. This can

Table 2: Brief description of functions in **accelerometry**, and whether each function uses C++ code.

Function	Description	C++
`accel.artifacts`	Replace extreme counts with mean of neighboring values.	Yes
`accel.bouts`	Identify bouts of physical activity. Can specify minimum count value, minimum length, and tolerance.	Yes
`accel.intensities`	Compute number of minutes with counts in various user-defined intensity levels and counts accumulated from each intensity level.	No
`accel.sedbreaks`	Count number of sedentary breaks.	Yes
`accel.weartime`	Identify periods when participant is not wearing the accelerometer.	Yes
`blockaves`	Calculate average of non-overlapping segments of data.	Yes
`movingaves`	Calculate moving averages.	Yes
`rle2`	Run length encoding. Often much faster than base function `rle`, and records value, length, and indices for each run.	Yes
`inverse.rle2`	Convert matrix created by `rle2` back to vector form.	No
`accel.process.uni`	Composite function to process uniaxial accelerometer data.	No
`accel.process.tri`	Composite function to process triaxial accelerometer data.	No

be done in ActiLife or in R using one of several available functions (e.g. `blockaves` in **accelerometry**, `aggAccFile` in **pawacc**, or `dataCollapser` in **PhysicalActivity**; Choi et al. 2011a).

Once the data is in R, the user should have a four-column matrix or data frame where the columns represent counts in the vertical axis, anteroposterior (AP) axis, and mediolateral (ML) axis, as well as steps. A sample matrix with this format is included in the **accelerometry** package. The next step is to install **accelerometry** and call `accel.process.tri`. The following code installs the package and processes the sample dataset in two ways: first using default settings, and then using a different algorithm for non-wear detection, and requesting a larger set of PA variables.

```
# Install and load accelerometry package
install.packages("accelerometry")
library("accelerometry")

# Load four-column matrix with counts and steps over 7 days
data(tridata)

# Generate basic PA variables using default settings
dailyPA1 <- accel.process.tri(counts.tri = tridata[, 1:3], steps = tridata[, 4])

# Request full set of PA variables, and use triaxial vector magnitude for non-wear
# detection rather than vertical axis, with 90-minute rather than 60-minute window
dailyPA2 <- accel.process.tri(counts.tri = tridata[, 1:3], steps = tridata[, 4],
                              brevity = 3, nonwear.axis = "mag", nonwear.window = 90)

# Check variable names for dailyPA1 and dailyPA2
colnames(dailyPA1)
colnames(dailyPA2)

# Print contents of dailyPA1 and first 15 variables in dailyPA2
dailyPA1
dailyPA2[, 1:15]

# Calculate average for cpm_vert from dailyPA1 and dailyPA2
mean(dailyPA1[, "cpm_vert"])
mean(dailyPA2[, "cpm_vert"])
```

Comparing the two datasets, we see that dailyPA1 has 15 variables for each of the seven days of monitoring, while dailyPA2 has 124. The brevity input was not specified in the first function call, so the default was used (brevity = 1). This resulted in only basic variables being generated: participant ID (id), day of week (day), whether the day was deemed valid for analysis (valid_day), wear time minutes (valid_min), total counts in each accelerometer axis (counts_vert, ..., counts_mag), average counts during wear time for each accelerometer axis (cpm_vert, ..., cpm_mag), and number of steps (steps). In contrast, setting brevity = 3 for dailyPA2 resulted in an additional 109 variables being generated. These extra variables quantify various aspects of daily activity that may be of interest to more experienced PA researchers. For example, one may want to test whether prolonged periods of sedentariness (sed_bouted_60min) are associated with health outcomes independent of total PA (cpm_vert), or compare daily patterns of PA across various demographics (cpm_hour1, ..., cpm_hour24). A data dictionary for these variables is available online (Van Domelen, 2013).

Looking at dailyPA1, it is interesting that counts_vert and counts_ap were very similar on all days except day 6, when counts_ap was more than three times greater than counts_vert. It seems likely that the participant engaged in some non-walking activity on this day that involved more anteroposterior (i.e. front-back) motion than vertical motion. This is supported by the participant having an unusually low number of steps on this day, but only slightly lower than average counts_ap.

A 60-minute non-wear algorithm based on only vertical-axis counts was used to create dailyPA1, while a 90-minute algorithm based on triaxial counts was used to create dailyPA2. The first algorithm identified some non-wear time on days 5 and 6, while the second non-wear algorithm did not mark any time on any day as non-wear. Some disagreement between different non-wear algorithms is to be expected, and different researchers may have different preferences based on validation studies or their own experiences processing accelerometer data. Regardless, it makes little difference which algorithm is used for this particular participant. Average counts per minute during wear time (cpm_vert) for the seven days of monitoring, a standard measure of total PA, was 142.7 using the first non-wear algorithm and 142.0 using the second.

Adding return.form = 1 to the function calls to accel.process.tri would result in daily averages being returned rather than separate values for each day of monitoring. This format is usually more useful for statistical analysis.

For further information on **accelerometry**, the package documentation includes a description of all of the functions and their inputs and has some additional examples (Van Domelen, 2014). The code from this section is also available in the package as a demo that can be executed by running demo("acceljournal").

R package 'nhanesaccel'

Overview

The package **nhanesaccel** is intended for researchers who want to use the objectively measured PA data in NHANES 2003–2006. Its primary function, nhanes.accel.process, converts minute-to-minute accelerometer data from NHANES participants into useful PA variables for statistical analysis.

Design goals

Design goals for **nhanesaccel** are shown in Table 3. Speed and flexibility were primary considerations for this package.

Implementation

The main function in **nhanesaccel** is nhanes.accel.process. This function relies heavily on the functions in **accelerometry**, and has similar inputs as accel.process.uni for controlling data processing methods. There are 31 function inputs controlling methods for non-wear detection, activity bouts, inclusion criteria, and the number of PA variables generated. These are described in the package documentation files (Van Domelen et al., 2014). A data dictionary for the variables generated by nhanes.accel.process is available online (Van Domelen, 2013).

The use of C++ code in the functions in **accelerometry** made it possible for nhanes.accel.process in **nhanesaccel** to process the full dataset of 14,631 participants in NHANES 2003–2006 in less than one minute. By including NHANES data in the package, there is no need for the user to download the large raw data files from the NHANES website. Adjusted sample weights, which are needed to ensure that the subset of participants with usable accelerometer data are weighted to match the demographics of the United States as a whole, are calculated by the function nhanes.accel.reweight. This function

Table 3: Design goals for R package **nhanesaccel**.

Goal	Description
1	Process NHANES 2003–2006 accelerometer data in less than 10 minutes.
2	Generate file for statistical analysis without directly modifying any code.
3	Produce .csv file that can be imported into any software package for analysis.
4	Do not require user to download 3.87 GB data files to the personal computer.
5	Calculate adjusted NHANES sample weights based on subset of participants with usable data, which are needed for appropriate statistical analysis.
6	Free.
7	Option to replicate methods used in the NCI's SAS programs, allowing researchers to use NCI's methods while taking advantage of goals 1–4 and 6.

operates very similarly to the NCI's SAS program reweight.pam (National Cancer Institute, 2007), and is called internally by nhanes.accel.process.

Version 2.1.1 of **nhanesaccel** is currently available on R-Forge. Due to its size (88.1 MB) it cannot be hosted on CRAN. The package requires R version 3.0.0 or above.

Examples

Users can install **nhanesaccel** via the install.packages function in R. Mac and Linux users and Windows users not running the most recent version of R (3.1 as of November 28, 2014) need to set type = "source" to install from source files rather than binaries.

```
# Install accelerometry package (if not already installed)
install.packages("accelerometry")

# Install nhanesaccel on Windows running most recent version of R
install.packages("nhanesaccel", repos = "http://R-Forge.R-project.org")

# Install nhanesaccel on Mac or Linux or on Windows running earlier version of R
install.packages("nhanesaccel", repos = "http://R-Forge.R-project.org",
                 type = "source")
```

Once the package is installed, the user can process the NHANES 2003–2006 dataset by loading the package and then calling nhanes.accel.process.

```
# Load nhanesaccel package
library("nhanesaccel")

# Process NHANES 2003-2006 data using default settings
nhanes1 <- nhanes.accel.process()

# Examine summary data for first 5 participants
nhanes1[1:5, ]
```

After a short period of time, the function returns a data frame named nhanes1 with daily averages for basic PA variables. Alternatively, the user may want to write a .csv file that can be imported into a different statistical software package for analysis. This can be achieved by adding the input write.csv = TRUE to the nhanes.accel.process function call. Unless otherwise specified via the directory input, the .csv file would be written to the current working directory, which can be seen by running getwd() and set by running setwd("<location>").

At this point the user could close R and switch to his or her preferred software package for statistical analysis. For example, the user could load the .csv file into SAS, merge in the Demographics files (which can be downloaded from the NHANES website), and use survey procedures in SAS to test for associations between demographic variables and PA.

Here we briefly illustrate typical data processing and analysis in R. Suppose we wish to test the hypothesis that American youth are more active on weekdays than on weekend days. First, we process the accelerometer data from NHANES 2003–2006 using three non-default inputs. The first two inputs require that participants have at least one valid weekday and one valid weekend day (rather than just one valid day overall), and the third requests that PA averages are calculated for weekdays and weekend days separately.

```
# Process NHANES 2003-2006 data, requiring at least one valid weekday and weekend day
nhanes2 <- nhanes.accel.process(valid.week.days = 1, valid.weekend.days = 1,
                                weekday.weekend = TRUE)

# Get dimension and variable names for nhanes2
dim(nhanes2)
names(nhanes2)
```

The data frame nhanes2 has 14,631 rows and 17 columns. Each row summarizes the PA of one NHANES participant. The first few variables are participant ID (seqn), NHANES year (wave; 1 for 2003–2004, 2 for 2005–2006), number of valid days of monitoring (valid_days, valid_week_days, and valid_weekend_days), and whether the participant has valid data for statistical analysis (include). Then we have daily averages for accelerometer wear time (valid_min), counts (counts), and counts per minute of wear time (cpm), for all valid days and for weekdays (wk_ prefix) and weekend days (we_ prefix) separately.

The variable cpm is a standard measure of total PA. We will compare weekday cpm (wk_cpm) and weekend cpm (we_cpm) in participants age 6 to 18 years to address our hypothesis. To get age for each participant, we merge a demographics dataset (included in **nhanesaccel**) to the nhanes2 data frame. Then we calculate for each participant the percent difference between wk_cpm and we_cpm. Positive values for cpm_diff indicate greater cpm on weekdays than on weekend days.

```
# Load demographics data and merge with nhanes2
data(dem)
nhanes2 <- merge(x = nhanes2, y = dem)

# Calculate percent difference between weekday and weekend CPM for each participant
nhanes2$cpm_diff <- (nhanes2$wk_cpm - nhanes2$we_cpm) /
                    ((nhanes2$wk_cpm + nhanes2$we_cpm) / 2) * 100
```

Now we are almost ready for statistical analysis. Because NHANES uses a complex multi-stage probability sampling design, analyzing the data as a simple random sample would result in incorrect inference. We need to use functions in the **survey** package (Lumley, 2014, 2004) rather than base R functions like t.test and glm. For those who are not familiar with concepts in survey analysis, reference materials and tutorials are available (Lumley, 2004, 2012, 2010).

Here we create a survey object using design features of NHANES.

```
# Create survey object called hanes
hanes <- svydesign(id = ~ sdmvpsu, strata = ~ sdmvstra, weight = ~ wtmec4yr_adj,
                   data = nhanes2, nest = TRUE)
```

Next we generate a figure comparing weekday and weekend PA in participants age 6 to 18 years. We use functions in the **survey** package to calculate mean (95% confidence interval) for cpm_diff in one-year age increments, then plot the results.

```
# Calculate mean (SE) and 95% CI's for cpm_diff for ages 6 to 18 years
mean.diff <- svyby(~ cpm_diff, by = ~ ridageyr, design = subset(hanes, ridageyr <= 18),
                   FUN = svymean, na.rm = TRUE)
ci <- confint(mean.diff)

# Plot means and CI's for cpm_diff by age
plot(x = 6:18, y = mean.diff[, 2], main = "CPM on Weekdays vs. Weekends",
     ylim = c(-30, 30), pch = 19, ylab = "Perc. diff. (mean +/- 95% CI)",
     xlab = "Age (years)", cex = 0.8, cex.axis = 0.85, cex.main = 1.5,
     cex.lab = 1.1, xaxt = "n")
axis(side = 1, at = 6:18, cex.axis = 0.85)
segments(x0 = 6:18, y0 = ci[, 1], x1 = 6:18, y1 = ci[, 2], lwd = 1.3)
abline(h = 0, lty = 2)
```

Figure 1 shows that PA is similar on weekdays and weekends from age 6 to 12 years, but 10–20% greater on weekdays from age 13 to 18 years (all $p < 0.05$). These results support our hypothesis that

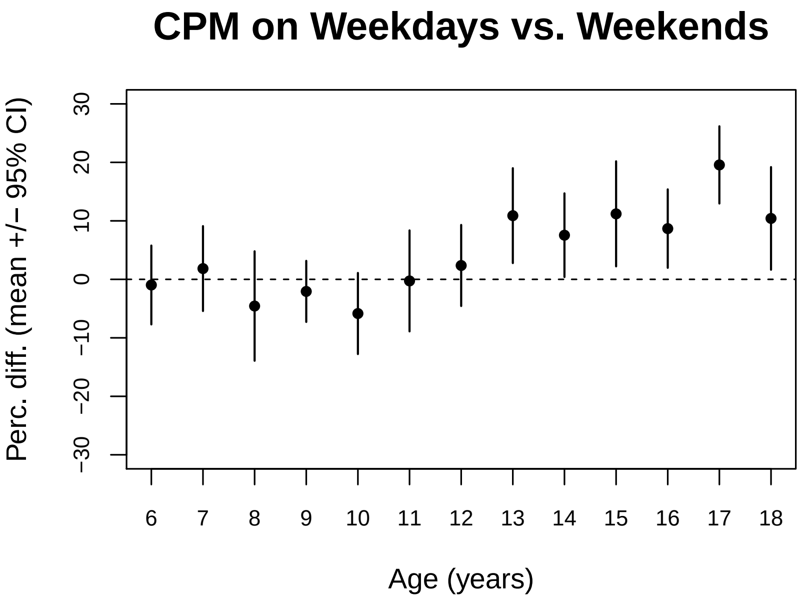

Figure 1: Mean (95% confidence interval) for percent difference between weekday and weekend CPM for NHANES 2003–2006 participants ages 6 to 18 years. Positive values indicate greater physical activity on weekdays.

American youth are more active on weekdays than on weekend days, but it appears that this is only the case for teenagers. Further analyses could try to identify the reason for the weekday/weekend difference in adolescents, for example by incorporating data on sports participation and time spent watching television for each age group.

Going back to the data processing step, users may wish to implement some non-default settings. To illustrate, the following code generates the full set of PA variables (203 variables), requires at least four valid days of monitoring, and uses a 90-minute rather than 60-minute minimum window for non-wear time.

```
# Process NHANES 2003-2006 data with four non-default inputs
nhanes3 <- nhanes.accel.process(brevity = 3, valid.days = 4, nonwear.window = 90,
                                weekday.weekend = TRUE)
```

The next example provides code to replicate the methods used in the NCI's SAS programs to process NHANES 2003–2006 data. In the first function call, we explicitly set each function input as needed to replicate the NCI's methods; in the second, we use the `nci.methods` argument as a shortcut.

```
# Specify count cut-points for moderate and vigorous intensity in youth
youthmod <- c(1400, 1515, 1638, 1770, 1910, 2059, 2220, 2393, 2580, 2781, 3000, 3239)
youthvig <- c(3758, 3947, 4147, 4360, 4588, 4832, 5094, 5375, 5679, 6007, 6363, 6751)

# Process NHANES 2003-2006 data using NCI's methods
nci1 <- nhanes.accel.process(waves = 3, brevity = 2, valid.days = 4,
                             youth.mod.cuts = youthmod, youth.vig.cuts = youthvig,
                             cpm.nci = TRUE, days.distinct = TRUE, nonwear.tol = 2,
                             nonwear.tol.upper = 100, nonwear.nci = TRUE,
                             weartime.maximum = 1440, active.bout.tol = 2,
                             active.bout.nci = TRUE, artifact.thresh = 32767,
                             artifact.action = 3)

# Repeat, but use nci.methods input for convenience
nci2 <- nhanes.accel.process(waves = 3, brevity = 2, nci.methods = TRUE)

# Verify that nci1 and nci2 are equivalent
all(nci1 == nci2, na.rm = TRUE)
```

Table 4: Mean (standard deviation) for time to process NHANES 2003–2004 data using **nhanesaccel** and using the NCI's SAS programs, based on five trials for each method.

nhanesaccel (defaults)	**nhanesaccel** (NCI methods)	NCI's SAS programs
8.64 (0.02) s	17.34 (0.04) s	25.03 (0.79) min

While we prefer using the default function inputs, there are some compelling reasons to use the NCI's methods instead. Doing so is simple, easy to justify in manuscripts, and ensures consistency with prior studies that have used the NCI's SAS programs. On the other hand, certain aspects of the NCI's methods are suboptimal. For example, the non-wear algorithm is applied to each day of monitoring separately, which results in misclassification of non-wear time as sedentary time when participants remove the accelerometer between 11 pm and midnight to go to sleep (Winkler et al., 2012). Of course, it is up to the user to decide whether to use the default settings, the NCI's methods, or some other combination of function inputs. Certainly many different approaches can be justified.

The code from this section of the paper is included in **nhanesaccel** as a demo that can be executed by running `demo("hanesjournal")`.

Processing times vs. NCI's SAS programs

One of the challenges of working with the NHANES accelerometer dataset is its size. A program written in R or MATLAB using standard loops to generate variables for nearly 15,000 participants can take hours to process. Efficiency was therefore a high priority in the development of **nhanesaccel**.

Processing times for **nhanesaccel** and the NCI's SAS programs were compared using R version 3.1.0 and SAS Version 9.4, respectively, on a Dell OptiPlex 990 desktop computer with Intel Core i7-2600 CPU at 3.4 GHz and 8 GB of RAM. The `system.time` function in R was used to record processing times for **nhanesaccel**, and "real time" output from the SAS log file was used for the NCI's programs. In both cases, elapsed times rather than CPU times were recorded. Results are shown in Table 4.

The package **nhanesaccel** was 174 times faster than the NCI's SAS programs under default settings, and 87 times faster using settings that replicate the NCI's methods. For **nhanesaccel**, the "NCI methods" settings have longer processing times than the defaults because the NCI's algorithms for non-wear and activity bout detection are more computationally intensive than the defaults.

Equivalence between **nhanesaccel** (NCI methods) and NCI's SAS programs was confirmed by direct comparison of the files generated. The only difference was that the NCI's SAS programs had data on nine participants with potentially unreliable data, whereas **nhanesaccel** excluded those participants entirely. Excluding this data is considered acceptable by the NCI (National Cancer Institute, 2007).

Both the NCI's SAS programs and **nhanesaccel** require some one-time steps prior to processing. The SAS programs require downloading and unzipping the data files, converting from .xpt format to SAS datasets, and downloading four SAS scripts and making minor changes to each. This process takes about ten minutes. As for **nhanesaccel**, installation can take a minute or two due to its large size.

Processing the full NHANES 2003–2006 using **nhanesaccel** takes approximately twice the time to process just NHANES 2003–2004. In five trials, mean (standard deviation) processing times were 17.57 (0.01) s using defaults and 40.26 (0.06) s using the NCI's methods.

Discussion

The **nhanesaccel** package should increase accessibility to the objectively measured PA data in NHANES 2003–2006. The software is free, efficient, and allows flexibility in data processing without writing original code to process over a billion data points. Proficiency in R is not required, as researchers can obtain a file for statistical analysis in SAS or another software package with just three lines of R code (install package, load package, call `nhanes.accel.process`). For researchers with data from studies other than NHANES, the functions in **accelerometry** should greatly simplify the process of converting time-series count data into meaningful PA variables.

There are several limitations of the packages presented in this article. First, the functions are not immediately useful for processing accelerometer data collected in less than one-minute epochs. Researchers with 1-s, 30-Hz, or 80-Hz data could convert the data to one-minute epochs and then use the functions in **accelerometry**, but doing so would sacrifice any additional information contained in the higher resolution data. Also, the default settings for the functions in **accelerometry** and **nhanesaccel** were chosen for data collected from accelerometers worn at the hip. Users who wish to process wrist data could still use the functions, but would have to take care to adjust function parameters, particularly `int.cuts`.

Another limitation is that the NHANES 2003–2006 dataset is 8–11 years old, and the data has been available for about six years. However, NHANES 2003–2006 remains the most recent study with objectively measured PA on a nationally representative sample of Americans. Although population activity levels may have changed since 2003–2006, associations between PA and outcomes should be relatively constant over time. Additionally, data on deaths will continue to be updated, which will enable future studies on PA and 5- or 10-year mortality. There are likely still many opportunities for useful contributions to the PA literature from this dataset.

There are several other R packages for processing accelerometer data. The **PhysicalActivity** package (Choi et al., 2011b) has functions for converting accelerometer data to longer epochs (`dataCollapser`), plotting accelerometer data (`plotData`), and implementing a non-wear algorithm (`wearingMarking`). Briefly, this algorithm uses a 90- rather than 60-minute window, and allows one or two non-zero count values provided that they are surrounded by 30 minutes of zero counts on both sides. Variants of this non-wear algorithm can be implemented by adjusting function inputs. The **GGIR** package (van Hees et al., 2014) has functions for processing high-frequency triaxial data from certain model accelerometers, and has a function for generating summary statistics. The variables that **GGIR** generates are mostly indicators of overall PA, such as mean acceleration over the full day or during time periods within the day. Finally, **pawacc** has functions for converting to longer epochs, detecting non-wear and activity bouts, generating summary statistics, and plotting data. The **pawacc** package can also be used to read accelerometer files into R without using ActiGraph's ActiLife software. To our knowledge, **nhanesaccel** is the only shared software currently available for processing the data from NHANES 2003–2006 other than the NCI's SAS programs.

Looking forward, there is a great need for software to process high-frequency accelerometer data. In particular, wrist-worn accelerometers are being used in NHANES 2011–2014, with the devices recording triaxial data at 80-Hz (Troiano, 2012). Analyzing this data will be challenging, as there will be over 145 million data points per participant, or about two trillion data points total. Aside from the sheer volume of data, methods for generating meaningful activity variables from high-frequency triaxial data are still being developed. It will be interesting to see how this data will be made available to the public, and whether the NCI will provide data processing software similar to the SAS programs for NHANES 2003–2006. Provided that the raw data is made available in some form, we intend to add functions to **nhanesaccel** or develop a separate R package to process NHANES 2011–2014 data.

We hope that **accelerometry** and **nhanesaccel** will make it easier for researchers to work with accelerometer data, and particularly the NHANES 2003–2006 dataset. Researchers who have any problems using either package, or have suggestions for additional features, should not hesitate to contact us.

Acknowledgments

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0940903.

Bibliography

ActiGraph. ActiLife 6 Data Analysis Software. Actigraph, Pensacola, Florida, United States. URL `http://www.actigraphcorp.com/product-category/software`, 2014. [p53]

A. Bankoski, T. B. Harris, J. J. McClain, R. J. Brychta, P. Caserotti, K. Y. Chen, D. Berrigan, R. P. Troiano, and A. Koster. Sedentary activity associated with metabolic syndrome independent of physical activity. *Diabetes Care*, 34(2):497–503, 2011. [p52]

V. Carson and I. Janssen. Volume, patterns, and types of sedentary behavior and cardio-metabolic health in children and adolescents: A cross-sectional study. *BMC Public Health*, 11(274), 2011. [p52]

L. Choi, Z. Liu, C. E. Matthews, and M. S. Buchowski. *PhysicalActivity: Process Physical Activity Accelerometer Data*, 2011a. URL `http://CRAN.R-project.org/package=PhysicalActivity`. R package version 0.1-1. [p54]

L. Choi, Z. Liu, C. E. Matthews, and M. S. Buchowski. Validation of accelerometer wear and nonwear time classification algorithm. *Medicine & Science in Sports & Exercise*, 43(2):357–364, 2011b. [p60]

L. Choi, S. C. Ward, J. F. Schnelle, and M. S. Buchowski. Assessment of wear/nonwear time classification algorithms for triaxial accelerometer. *Medicine & Science in Sports & Exercise*, 44(10):2009–2016, 2012. [p52, 53]

R. C. Colley, D. Garriguet, I. Janssen, C. L. Craig, J. Clarke, and M. S. Tremblay. Physical activity of Canadian children and youth: Accelerometer results from the 2007 to 2009 Canadian Health Measures Survey. *Health Reports*, 22(1):1–9, 2011. [p52]

J. L. Copeland and D. W. Esliger. Accelerometer assessment of physical activity in active, healthy older adults. *Journal of Aging and Physical Activity*, 17(1):17–30, 2009. [p52]

D. Eddelbuettel. *Seamless R and C++ Integration with Rcpp*. Springer, New York, 2013. ISBN 978-1-4614-6867-7. [p53]

D. Eddelbuettel and R. François. Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, 40(8):1–18, 2011. URL http://www.jstatsoft.org/v40/i08/. [p53]

M. Geraci. *pawacc: Physical Activity with Accelerometers*, 2014. URL http://CRAN.R-project.org/package=pawacc. R package version 1.2. [p53]

M. Geraci, C. Rich, F. Sera, M. Cortina-Borja, L. J. Griffiths, and C. Dezateux. Technical report on accelerometry data processing in the Millennium Cohort Study. London, UK: University College London, 2012. URL http://discovery.ucl.ac.uk/1361699. [p53]

M. Hagstromer, P. Oja, and M. Sjostrom. Physical activity and inactivity in an adult population assessed by accelerometry. *Medicine & Science in Sports & Exercise*, 39(9):1502–1508, 2007. [p52]

G. N. Healy, C. E. Matthews, D. W. Dunstan, E. A. Winkler, and N. Owen. Sedentary time and cardio-metabolic biomarkers in U.S. adults: NHANES 2003–2006. *European Heart Journal*, 32(5): 590–597, 2011. [p52]

T. Lumley. Analysis of complex survey samples. *Journal of Statistical Software*, 9(1):1–19, 2004. URL http://www.jstatsoft.org/v09/i08/. [p57]

T. Lumley. *Complex Surveys: A Guide to Analysis Using R*. John Wiley & Sons, Inc., Hoboken, NJ, 2010. [p57]

T. Lumley. Survey analysis in R. Website for 'survey' package. URL http://R-survey.R-Forge.R-project.org/survey, 2012. [p57]

T. Lumley. *survey: Analysis of Complex Survey Samples*, 2014. URL http://CRAN.R-project.org/package=survey. R package version 3.30. [p57]

National Cancer Institute. Risk factor monitoring and methods: SAS programs for analyzing NHANES 2003–2004 accelerometer data. URL http://riskfactor.cancer.gov/tools/nhanes_pam, 2007. [p52, 56, 59]

National Instruments. *LabVIEW: Laboratory Virtual Instrument Engineering Workbench*. Austin, Texas, United States, 2014. URL http://www.ni.com/labview. [p52]

S. B. Sisson, S. M. Camhi, T. S. Church, C. Tudor-Locke, W. D. Johnson, and P. T. Katzmarzyk. Accelerometer-determined steps/day and metabolic syndrome. *American Journal of Preventive Medicine*, 38(6):575–582, 2010. [p52]

The MathWorks, Inc. *MATLAB – The Language of Technical Computing, Version R2014b*. The MathWorks, Inc., Natick, Massachusetts, 2014. URL http://www.mathworks.com/products/matlab/. [p52]

R. P. Troiano. Physical activity among children and adolescents: Data from National Health and Nutrition Examination Survey (NHANES) 2003-2006. URL http://www.cdc.gov/nchs/ppt/nchs2012/SS-15_TROIANO.pdf, 2012. [p60]

R. P. Troiano, D. Berrigan, K. W. Dodd, L. C. Mâsse, T. Tilert, M. McDowell, et al. Physical activity in the United States measured by accelerometer. *Medicine & Science in Sports & Exercise*, 40(1):181, 2008. [p52]

C. Tudor-Locke, M. M. Brashear, W. D. Johnson, and P. T. Katzmarzyk. Accelerometer profiles of physical activity and inactivity in normal weight, overweight, and obese U.S. men and women. *The International Journal of Behavioral Nutrition and Physical Activity*, 7(60), 2010. [p52]

C. Tudor-Locke, S. M. Camhi, and R. P. Troiano. A catalog of rules, variables, and definitions applied to accelerometer data in the National Health and Nutrition Examination Survey, 2003-2006. *Preventing Chronic Disease*, 9:E113, 2011. [p52]

D. R. Van Domelen. Personal/professional website. URL http://sites.google.com/site/danevandomelen, 2013. [p53, 55]

D. R. Van Domelen. *accelerometry: Functions for Processing Uniaxial Minute-to-Minute Accelerometer Data*, 2014. URL http://CRAN.R-project.org/package=accelerometry. R package version 2.2.4. [p52, 55]

D. R. Van Domelen, W. S. Pittard, and T. B. Harris. *nhanesaccel: Process Accelerometer Data from NHANES 2003–2006*, 2014. URL http://R-Forge.R-project.org/projects/nhanesaccel/. R package version 2.1.1. [p52, 55]

V. T. van Hees, Z. Fang, and J. H. Zhao. *GGIR: A Package to Process Multi-Day Raw Accelerometer Data*, 2014. URL http://CRAN.R-project.org/package=GGIR. R package version 1.0-1. [p60]

E. A. H. Winkler, P. A. Gardiner, B. K. Clark, C. E. Matthews, N. Owen, and G. N. Healy. Identifying sedentary time using automated estimates of accelerometer wear time. *British Journal of Sports Medicine*, 46:436–442, 2012. [p59]

Dane R. Van Domelen
Department of Biostatistics and Bioinformatics
Rollins School of Public Health
Emory University
1518 Clifton Road, Room 323
Atlanta, GA 30322
United States
dvandom@emory.edu

W. Stephen Pittard
Department of Biostatistics and Bioinformatics
Rollins School of Public Health
Emory University
1518 Clifton Road, Room 366
Atlanta, GA 3022
United States
wsp@emory.edu