Clustered Recognition of Interrupted Bouts (CRIB)

Paul R. Hibbing

Introduction and Installation

This vignette will show you how to implement the CRIB method. The first step is making sure you have the `PBpatterns` package installed on your computer. Here's how:

```
## `remotes` is a package that makes it easy to install packages from GitHub, but
## in my experience it sometimes struggles to install the related packages (i.e.,
## dependencies) correctly. So first we'll do a manual workaround. All it's
## doing is looking through a list of required packages, and installing any of
## them that haven't already been installed (they'll be skipped if they have).
## Be aware: Some of these packages may have long installation times.

invisible(lapply(
  c(
    "DescTools", "ggplot2", "magrittr", "PAutilities", "purrr", "utils",
    "AGread", "PhysicalActivity", "tree", "randomForest", "knitr", "rmarkdown"
  ),
  function(x) if (!x %in% installed.packages()) install.packages(x)
))

## Once that's done, we can (hopefully) install from GitHub
remotes::install_github("paulhibbing/PBpatterns", dependencies = FALSE)
```

Copy and paste the above into your R console, then hit enter to run it.

Preparation

Once you have the package installed, all you need is some activity data and the `analyze_bouts` function. For this demonstration, let's use some sample NHANES data.

```
data(example_data, package = "PBpatterns")
```

This dataset has activity counts that we can use to look at bouts of moderate-to-vigorous physical activity (MVPA). For illustration, let's say we initially coded our data as sedentary behavior ($PAXINTEN \leq 100$), light physical activity ($PAXINTEN\ 101 - 759$), or MVPA ($PAXINTEN \geq 760$).

```
## Determine minute-by-minute intensity
intensity <- cut(
  example_data$PAXINTEN,
  breaks = c(-Inf, 101, 760, Inf),
  labels = c("SB", "LPA", "MVPA"),
  right = FALSE
)
```

Running the Code

Once we have our data (`intensity` in this case), we can plug it into `analyze_bouts`. Let's see the code first, then go over what it means.

```
mvpa_bouts <- PBpatterns::analyze_bouts(
  x = intensity,
  target = "MVPA",
  method = "CRIB",
  target_buffer_mins = 10,
  longest_allowable_interruption_mins = 2,
  required_percent = 80,
  max_n_interruptions = Inf,
  minimum_bout_duration_minutes = 10,
  epoch_length_sec = 60
)
```

Here is what each piece means:

- **mvpa_bouts <-** Store the function results in an object called `mvpa_bouts`
- **PBpatterns::analyze_bouts** This tells R to find the `analyze_bouts` function in the `PBpatterns` package. In fact, if you run `PBpatterns::analyze_bouts` in your console, R will print the source code.
- **x = intensity** Here we specify that our input datastream is `intensity`, as defined in the earlier code.
- **target = "MVPA"** Here we specify which behavior we are interested in. The input data (`intensity`) has values in the set {SB, LPA, MVPA}, and we would like to look specifically at bouts of `MVPA`, with the other behaviors being lumped together in a single group called `other`.
- **method = "CRIB"** Here we specify that R should run the CRIB method. To explore the other options, check out the Bout Analysis and Sedentary Patterns vignette.
- **target_buffer_mins = 10** Here we specify how our data should be stratified/partitioned. In this case, `intensity` will be stratified/partitioned anytime we see ≥ 10 consecutive minutes of `other` behavior.
- **longest_allowable_interruption_mins = 2** Here we specify that a valid bout should not include any single interruption lasting longer than 2 minutes.
- **required_percent = 80** Here we specify that a valid bout should be interrupted for no more than 20% of its full duration.
- **max_n_interruptions = Inf** Here we specify that a valid bout can have unlimited interruptions as long as the criteria for `longest_allowable_interruption_mins` and `required_percent` are met.
- **minimum_bout_duration_minutes = 10** Here we specify that only bouts lasting ≥ 10 min should be included in the output.
- **epoch_length_sec = 60** Here we specify what the epoch length of `intensity` is.

The above elements are set up to allow flexible bout criteria depending on the research question and the variable of interest. In our example, we set `max_n_interruptions = Inf` to avoid a restriction in that area – similar approaches can be taken for other settings as well, by setting them to `0` or `Inf` as appropriate. (For `required_percent`, 100 is the upper limit rather than `Inf`.) Notably, `minimum_bout_duration_minutes` is a filtering criterion. It has no direct effect on how the bouts are defined; it simply affects which ones are retained *after* they have been defined.

Interpreting the Output

Now let's take a look at the output and go over what it means:

```
mvpa_bouts
#>   start_index end_index values n_total_events n_value_events
#> 1         912       932   MVPA              7              4
#> 2         998      1011   MVPA              5              3
#> 3        1022      1039   MVPA              1              1
#> 4        1042      1063   MVPA              5              3
#> 5        2212      2221   MVPA              1              1
#> 6        3910      3922   MVPA              3              2
#> 7        4166      4188   MVPA              5              3
#>   n_interruption_events overall_minutes engaged_minutes percent_time_engaged
#> 1                     3              21              17             80.95238
#> 2                     2              14              12             85.71429
#> 3                     0              18              18            100.00000
#> 4                     2              22              20             90.90909
#> 5                     0              10              10            100.00000
#> 6                     1              13              12             92.30769
#> 7                     2              23              21             91.30435
#>   total_interruption_minutes longest_interruption_minutes
#> 1                          4                            2
#> 2                          2                            1
#> 3                          0                            0
#> 4                          2                            1
#> 5                          0                            0
#> 6                          1                            1
#> 7                          2                            1
```

This is a data frame with one row per bout. The variables are:

- **start_index** The starting point of the bout (e.g., `intensity[912]` for the first bout in this example)
- **end_index** The ending point of the bout (e.g., `intensity[932]` for the first bout in this example)
- **values** A meaningless constant (equal to the setting of `target`), left over from run length encoding
- **n_total_events** The number of distinct behavior events occurring between `start_index` and `end_index`
- **n_value_events** The number of distinct target behavior events occurring between `start_index` and `end_index` (referred to as `value` events in reference to the `values` column)
- **n_interruption_events** The number of distinct interruption events occurring between `start_index` and `end_index`
- **overall_minutes** The combined duration of all `value` and `interruption` events
- **engaged_minutes** The combined duration of all `value` events
- **percent_time_engaged** Percentage of `length_total` comprised by `length_value`
- **total_interruption_minutes** The combined duration of all `interruption` events
- **longest_interruption_minutes** The duration of the single longest interruption event

Expanding the Output

In some cases we may want to convert our bout information back to the original length of the input (i.e., `intensity`). We can use the `expand_bouts` function to accomplish that.

```
expanded <- PBpatterns::expand_bouts(mvpa_bouts)
str(expanded)
#>  Factor w/ 3 levels "other","MVPA",..: 1 1 1 1 1 1 1 1 1 1 ...
table(expanded)
#> expanded
#>        other         MVPA interruption
#>         9959          110           11
```

And we can also append that new variable into our original dataset as well.

```
example_data$intensity <- expanded
head(example_data)
#>     SEQN PAXSTAT PAXCAL PAXDAY PAXN PAXHOUR PAXMINUT PAXINTEN intensity
#> 1 21010       1      1      7    1       0        0        1     other
#> 2 21010       1      1      7    2       0        1        0     other
#> 3 21010       1      1      7    3       0        2        0     other
#> 4 21010       1      1      7    4       0        3        0     other
#> 5 21010       1      1      7    5       0        4        0     other
#> 6 21010       1      1      7    6       0        5        0     other
```

Plotting the output

If you want to visualize the results of your bout analysis, you can use a basic `plot` function.

```
plot(mvpa_bouts)
```
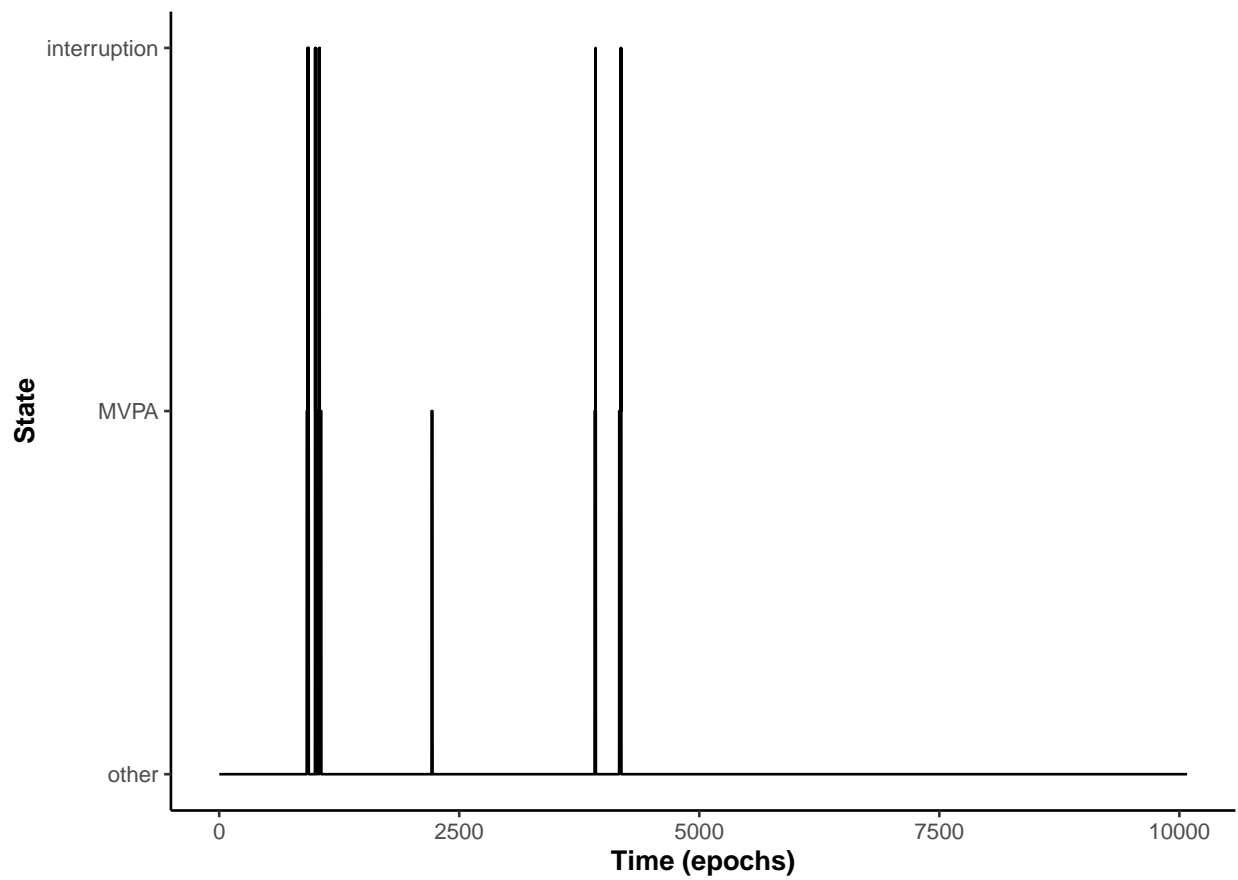
Conclusion

This should give you a broad sense of how to use the CRIB method and what else you can do with it. Feel free to post an issue on the GitHub page if any of the above gives you trouble. Good luck!