`cesR`: An R package for the Canadian Election Study*

Paul A. Hodgetts† Rohan Alexander‡

31 August 2020

Abstract

In this paper we introduce `cesR`, which is an R package that allows users to more easily use the Canadian Election Study (CES). The package has functions that download the CES surveys, add the questions that were asked, quickly preview a CES survey, and finally to obtain an opinionated subset of the 2019 survey. Our package is useful in both teaching and research by enhancing both reproducibility and accessibility.

1 Introduction

An election season brings with it a stream of predictions made using a variety of resources and methods. While *ex ante* predictions can provide insight as to the direction of an election and voter intention, *ex post* analysis of an election can be equally enlightening. Analysis of this sort can provide a better understanding of an election after-the-fact and can provide clarity as to the changes in the political climate over the years. The Canadian Election Study (CES) is a series of large-scale surveys that have been conducted in the lead-up to, and immediately after, each Canadian federal election since 1965. The CES series provides insight into such topics as the intentions of voters, what issues voters deem important, and the perception of parties and candidates (Canadian Election Study, 2019). For instance, Bittner (2018) used CES data to assess the level of presidentialization experienced in Canadian elections over time; while other studies have used CES data to assess the level of Islamophobia in Canada (Wilkins-Laflamme, 2018), how anti-party rhetoric affects voter behaviour (Gidengil et al., 2001), and how political trust influences voter behaviour (Belanger and Nadeau, 2005), among many other topics. The collection of, and access to, the CES data is integral to this research. The R package `cesR` ('caesar'), that is presented in this paper, seeks to complement the work of the CES, and enhance both research and teaching using this dataset. It does

*We thank Jean Jamieson-Hanes and Monica Howlett for their assistance. Our code and datasets are available at: https://github.com/hodgettsp/cesR and https://github.com/hodgettsp/ces_data. Comments on the 31 August 2020 version of this paper are welcome at: rohan.alexander@utoronto.ca. For the latest version please click here.

†Faculty of Information, University of Toronto.

‡Faculty of Information and Department of Statistical Sciences, University of Toronto, rohan.alexander@utoronto.ca.

this by providing a means for users to easily access CES survey datasets in a reproducible manner.

The CES survey datasets are publicly available from various repositories across the internet in various data types (Canadian Election Study, 2019; UBC, 2015; ODESI, 2020). The `cesR` package brings together these various sources, providing users with a means of easily accessing the universe of CES surveys while also circumventing the need to download and store survey data files. The structure of `cesR` also removes the need to call a data file from a directory, thereby making it easier to work with CES datasets between different computing environments and for easier for beginners. Additionally, `cesR` provides a function, `get_decon()`, that creates a subset of the 2019 CES online survey that has been prepared in an opinionated way. This subset of the 2019 CES online survey provides a resource for teachers who want to use real-world datasets in their teaching.

The `cesR` package is built using the R programming language and is designed to work within the R integrated development environment (IDE) RStudio (R Core Team, 2020; RStudio Team, 2020). Providing access to data is common in R packages and the `cesR` package follows and was inspired by similar packages such as `opendatatoronto` (Gelfand, 2020), `Lahman` (Friendly et al., 2020), `fueleconomy` (Wickham, 2020), and `nasaweather` (Wickham, 2014). Furthermore, the `cesR` package is complementary to packages such as `dataverse` (Leeper, 2017) and `cancensus` (von Bergmann et al., 2020) that provide users access to Canadian survey and census data. Packages such as these enhance reproducibility as they allow more of the user workflow to be documented in code. They also enhance the teaching of these initial steps of the data analysis workflow by allowing a consistent and repeatable approach.

This paper introduces the `cesR` package. Section 2 discusses the functions: `get_ces()`, `get_cescodes()`, `get_question()`, `get_preview()`, and `get_decon()`. Section 3 provides an example of the package usage. And finally, Section 4 discusses plans for the package and aspects to be aware of.

2 Package Functions

The `cesR` package has five functions that allow users to download and explore the CES survey datasets. CES survey datasets can be downloaded using either the `get_ces()` function or the `get_preview()` function. Where `get_ces()` provides users with complete CES survey datasets, `get_preview()` provides truncated datasets that may be more useful for initial exploratory analysis. Datasets can be further explored using the `get_question()` function, which returns the survey question associated with a given column in a given dataset. In all of these functions, CES survey datasets are called using designated character string arguments. The function `get_cescodes()` provides users a convenient means of looking up these arguments. Finally , the `get_decon()` function provides users with a pre-created subset of the 2019 CES online survey that could be especially useful in teaching.

2.1 The `get_ces()` function

The `get_ces()` function returns a requested CES survey as a data object and prints to the console the associated citation and URL for the survey dataset repository. The function takes one argument in the form of a character string. This argument is a vector member that has been associated with a CES survey through the body of code in the `get_ces()` function. When used it calls the download URL for the survey from an associated GitHub repository named `ces_data` (https://github.com/hodgettsp/ces_data). If the provided character string argument matches a member of the built-in vector `ces_codes`, the associated file is downloaded using the `download.file()` function from the `utils` R package (R Core Team, 2020). This file is downloaded as a compressed .zip folder and is stored temporarily in an `inst/extdata` directory in the user's working directory. After it is downloaded the compressed folder is unzipped using the `unzip()` function from the `utils` R package (R Core Team, 2020) and read into R using either the `read_dta()` function or the `read_sav()` function from the `haven` R package (Wickham and Miller, 2020) depending on the file extension of the downloaded file. A data frame is then assigned using the `assign()` function from `base` R (R Core Team, 2020) as a data object in the global environment. The downloaded file and file directory are then removed from the computer using the `unlink()` function from `base` R (R Core Team, 2020). Finally, the recommended citation for the requested survey dataset and URL of the survey data storage location are printed in the console.

```
# install the cesR package from GitHub
devtools::install_github("hodgettsp/cesR")

# load the cesR package into RStudio
library(cesR)

# call the 2019 CES online survey
get_ces("ces2019_web")
```

If the provided character string argument does not have a match in the built-in vector, then the function process is stopped and the warning message `'Error in get_ces(): Warning: Code not in table'` is printed in the RStudio console.

```
# install the cesR package from GitHub
devtools::install_github("hodgettsp/cesR")

# load the cesR package into RStudio
library(cesR)

# call the 2019 CES online survey (note the typo: it should be "ces2019_web" as above)
get_ces("2019ces_web")
```

The character string arguments calls for each CES survey are provided in the `get_cescodes()` function discussed in Section 2.2.

The structure of the `get_ces()` function makes it possible to call a CES survey more than

once, but doing so will re-create the data object. When the `get_ces()` function is called, before downloading the requested survey file, the `get_ces()` function checks if the file already exists in the `inst/extdata` directory. While the `get_ces()` function is designed to remove the downloaded file, checking if the file already exists alerts the function if something is wrong. By checking if the file exists and not if the data object exists, the `get_ces()` function is able to load the requested dataset more than once, thereby allowing an unmanipulated version of a dataset to be loaded if required. For instance, if a loaded data object becomes corrupted or an unmanipulated version is needed, then using `get_ces()` to call the same CES survey will provide a clean copy.

CES survey dataset files use either the `.dta()` or `.sav()` file extension, meaning the datasets are loaded into R and RStudio as one of these types. But by default `get_ces()` function does not convert the values of the loaded tables to a factor type so that a user's workflow is not interrupted and to reduce the chance of silent issues. However it is suggested that the `to_factor()` function from the `labelled` package (Larmarange, 2020) be used to convert the dataset values to a factor type.

```
# install cesR package from GitHub
devtools::intall_github("hodgettsp/cesR")

# install labelled package from CRAN
install.packages("labelled")

# load cesR and labelled packages
library(cesR)
library(labelled)

# request 2019 CES online survey
get_ces("ces2019_web")

# convert dataframe values to factor type
ces2019_web <- to_factor(ces2019_web)

# check column heads for dataframe
head(ces2019_web)
```

2.2 The `get_cescodes()` function

The `get_cescodes()` function provides a user with a means of looking up the arguments that `get_ces()` uses to access each CES survey dataset. The `get_cescodes()` function does not take any arguments. Instead when the function is called it prints to the console a dataframe that contains the survey codes and their associated argument calls. See Section 3 for an example of the `get_cescodes()` function used as part of the full `cesR` package.

```
# install cesR package from GitHub
devtools::install_github("hodgettsp/cesR")
```

Table 1: CES surveys and their codes

Index	CES survey	cesR survey code
1	2019 web	ces2019_web
2	2019 phone	ces2019_phone
3	2015 web	ces2015_web
4	2015 phone	ces2015_phone
5	2015 combo	ces2015_combo
6	2011	ces2011
7	2008	ces2008
8	2004	ces2004
9	2004-2011	ces0411
10	2004-2006	ces0406
11	2000	ces2000
12	1997	ces1997
13	1993	ces1993
14	1988	ces1988
15	1984	ces1984
16	1974	ces1974
17	1974 & 1979 & 1980	ces7480
18	1972 Jun/Jul	ces72_jnjl
19	1972 Sep	ces72_sep
20	1972 Nov	ces72_nov
21	1968	ces1968
22	1965	ces1965

```
# load cesR package into RStudio
library(cesR)

# get CES survey argument calls
get_cescodes()
```

The possible codes are contained in Table 1.

The `get_cescodes()` function works by constructing two vectors, one vector contains the CES survey codes and the other contains the associated survey argument calls. The function then creates dataframes of two vectors using the `data.frame()` function from `base` R and adds an index number column using the `seq()` function from `base` R (R Core Team, 2020) by which to merge the dataframes. Using the `merge()` function from `base` R (R Core Team, 2020), the dataframes are then merged by the index number into a new dataframe. Using the index number ensures the dataframes remain ordered and merge correctly across rows.

Column names in the new dataframe are then renamed using the `rename()` function from the `dplyr` package (Wickham et al., 2020a) and the vector objects are removed from the environment. The `get_cescodes()` function does not create any variable that is available in the global environment.

2.3 The `get_question()` function

The `get_question()` function provides users with the ability to look up a survey question associated with a given column name. The function takes two arguments in the form of character strings. These are the name of a data object and the name of a column in the given data object. The function works such that it checks whether the given data object exists using the `exists()` function from `base` R (R Core Team, 2020). If the given column exists in the given data object, `get_question()` will print the variable label of the given column to the console using a combination of the `var_label()` function from the `labelled` package (Larmarange, 2020) and the `get()` function from `base` R (R Core Team, 2020). See Section 3 for its use as part of the whole `cesR` package.

```
# install cesR package from GitHub
devtools::install_github("hodgettsp/cesR")

# load package into library
library(cesR)

# load 2019 phone dataset
get_ces("ces2019_phone")

# get question for column q11
get_question("ces2019_phone", "q11")
```

If the object does not exist, the function will print the warning in the console `'Warning: Data object does not exist'`. If the object does exist, `get_question()` will check if the given column name exists in the given data object. This is done using a combination of the `hasName()` function from the `utils` package and the `get()` function from `base` R (R Core Team, 2020). The `hasName()` function checks if the given column name is in the given data object. Because the arguments are given as character strings the `get()` function is used to return the actual data object instead of the provided character string. Otherwise, the `hasName()` function would only check if the given column name argument occurred in the given character string argument and not the actual data object.

```
# install cesR package from GitHub
devtools::install_github("hodgettsp/cesR")

# load package into library
library(cesR)

# load 2019 phone dataset
```

```
get_ces("ces2019_phone")

# get question for column q11 (for an object that does not exist)
get_question("2019_phone", "q11")
```

If the column does not exist in the data object, the following warning is printed in the console: `'Warning: Variable is not in dataset'`.

```
# install cesR package from GitHub
devtools::install_github("hodgettsp/cesR")

# load package into library
library(cesR)

# load 2019 phone dataset
get_ces("ces2019_phone")

# get question for column q11 (for a column that does not exist)
get_question("ces2019_phone", "11")
```

In addition to being usable with the data objects created from the `get_ces()` function, the `get_question()` function is structured in such a way that it can also be used to return the column label for the `decon` dataset or any dataset of the labelled type.

```
# install cesR package from GitHub
devtools::install_github("hodgettsp/cesR")

# load package into library
library(cesR)

# load decon dataset
get_decon()
head(decon)

# get question for education column
get_question("decon", "education")
```

2.4 The `get_preview()` function

To make the initial data exploration phase easier it can be useful to have access to a truncated version of a dataset. Additionally, a truncated dataset provides a resource that can be used when teaching exploratory data analysis. The `get_preview()` function provides such truncated versions of the CES datasets. The function takes two arguments, a character string to call a survey of the same style as used for the `get_ces()` function and a numerical value that sets the number of rows returned.

```
#install cesR package
devtools::install_github("hodgettsp/cesR")

# load package into library
library(cesR)

# get preview for 2019 CES online survey for 10 rows
get_preview("ces2019_web", 10)
```

In the initial argument classifications for the `get_preview()` function the variable `x` is set to equal six. This sets a default value for the number of rows to be returned if this value is left empty while allowing the value to be overwritten if provided a value.

```
# install cesR package
devtools::install_github("hodgettsp/cesR")

# load package into library
library(cesR)

# get preview for 2019 CES online survey
get_preview("ces2019_web")
```

Like the `get_ces()` function, the `get_preview()` function checks if a file exists in the cesR directory before downloading the requested survey data file associated with the given character string argument. If the file does not already exist, the survey data file is downloaded using the `download.file()` function from `base` R to a temporary directory set using the `tempfile()` function also from `base` R (R Core Team, 2020). The downloaded file is then unzipped into the `cesR` package directory using the `unzip()` function from the `utils` package (R Core Team, 2020). The extracted survey data file is read into R in the `get_preview()` function using either the `read_dta()` or `read_sav()` functions from the `haven` package depending upon the file extension (Wickham and Miller, 2020). The `get_preview()` function then creates a truncated version of the requested dataset by converting the dataset values to a factor type using the `to_factor()` function from the `labelled` package (Larmarange, 2020) and truncating the dataset using the `head()` function from the `utils` package (R Core Team, 2020) by the numeric value provided by the argument `x`. This dataset is assigned as a variable with the suffix `_preview` to the global environment using the `assign()` function from `base` R (R Core Team, 2020). The function then removes the downloaded files using the `unlink()` function from `base` R and removes the local variable holding the full dataset using the `rm()` function also from `base` R (R Core Team, 2020). See Section 3 for a use of the `get_preview()` function in conjunction with the other `cesR` functions.

2.5 The `get_decon()` function

When called, the `get_decon()` function creates a subset of the 2019 CES online survey under the name `decon` (demographics and economics) that provides a tool for teaching survey

datasets and their analysis. The `get_decon()` function takes no arguments. The function first checks the global environment to see if an object named `decon` exists using the `exists()` function from `base` R (R Core Team, 2020). This prevents the `decon` dataset from being recreated if the object exists. If the `get_decon()` function is run when an object with the name `decon` already exists a warning will print in the console `'Error in get_decon() : Warning: File already exists.'`

```
# install cesR package
devtools::install_github("hodgettsp/cesR")

# load cesR package
library(cesR)

# load decon dataset
get_decon()
head(decon)

# reload decon
get_decon()
```

If a situation arises in which the `decon` dataset needs to be recreated, then the best course of action is to use the `rm()` function from `base` R (R Core Team, 2020) to remove the `decon` object and then run the `get_decon()` function again.

```
# install cesR package
devtools::install_github("hodgettsp/cesR")

# load cesR package
library(cesR)

# load decon dataset
get_decon()
head(decon)

# remove decon object
rm(decon)

# relaoad decon
get_decon()
```

After checking if an object with the name `decon` exists, the `get_decon()` function performs similarly to the `get_ces()` function. It assigns a temporary file extension with a .zip type using the `tempfile()` function from `base` R, and calls on the URL for the 2019 CES online survey and temporarily downloads the associated file using the `download.file()` function from `base` R (R Core Team, 2020). After unpacking the compressed folder with the `unzip()` function from the `utils` package (R Core Team, 2020), the file is read into R using the `read_dta()`

function from the `haven` package (Wickham and Miller, 2020). Using the `select()` function from the `dplyr` package, 20 variables are selected from the main CES 2019 online survey and renamed using the `rename` function from the `dplyr` package (Wickham et al., 2020a). The dataframe values are then converted to a factor type using the `to_factor()` function from the `labelled` package (Larmarange, 2020). A new variable, consisting of participants' responses to their self-perceived position on the political spectrum, is then created from two variables using the `unite()` function from the `tidyr` package (Wickham and Henry, 2020) and all empty cells are replaced with `NA` values using the `mutate()` function from the `dplyr` package (Wickham et al., 2020a). The dataset is made available in the global environment by using the `assign()` function from `base` R to create a new data object under the name `decon`. Finally, the downloaded files are removed from the computer using the `unlink()` function from `base` R (R Core Team, 2020) and a citation for the 2019 CES online survey and a URL to the storage location is printed to the console.

```
# install cesR package
devtools::install_github("hodgettsp/cesR")

# load cesR package into RStudio
library(cesR)

# call decon dataset
get_decon()
head(decon)
```

The columns in the `decon` dataset are contained in Table 2.

3 Vignette

3.1 Creating a subset of the CES 2019 phone survey dataset

While the `get_decon()` function provides a subset of the CES 2019 online survey dataset, the general use of the `cesR` package is to access CES data and the subsetting of any of the CES survey datasets.

The following presents a vignette of installing `cesR` from its GitHub repository as well as calling and producing a subset of the 2019 CES phone survey dataset. This vignette uses functions from the `cesR`, `devtools` (Wickham et al., 2020b), `labelled` (Larmarange, 2020), `dplyr` (Wickham et al., 2020a), and `utils` (R Core Team, 2020) packages.

To begin, install and load the `cesR` package (and all other necessary packages) into RStudio. Currently, this is only available through the use of the `install_github()` function from the `devtools` package (Wickham et al., 2020b). During installation, RStudio may request to update other packages.

```
# uncomment any package that needs to be installed
# install.packages("devtools")
# install.packages("labelled")
```

Table 2: Variables in the decon dataset and their description

decon survey variables	CES variable
citizenship	'Canadian citizenship status'
yob	'year of birth'
gender	'identified gender of the respondent'
province_territory	'Province or Territory of current residence'
education	'highest level of education completed'
lr	'united column of lr_bef and lr_aft values; whether the respondent identifies on the political spectrum'
lr_bef	'where the respondent identifies on the political spectrum; asked before party identification questions'
lr_aft	'where the respondent identifies on the political spectrum; asked after party identification questions'
religion	'religion of respondent'
sexuality_selected	'sexual identity'
sexuality_text	'sexual identity; written answers'
language_eng	'language learned as child and still understand; selected response English'
language_fr	'language learned as a child and still understand; selected response French'
language_abgl	'language learned as a child and still understand; specified Aboriginal language'
employment	'employment status'
income	'total household income before taxes for the year 2018'
income_cat	'selected household income category'
marital	'marital status'
econ_retro	'response to question: 'over the past year
econ_fed	'response to question: 'have the policies of the federal government made Canada's economy..."
econ_self	'response to question: have the policies of the federal government made your financial situation..."

```
# install.packages("dplyr")
# install.packages("tidyr")

# install cesR package from GitHub
devtools::install_github("hodgettsp/cesR")

library(cesR)
library(labelled)
library(dplyr)
```

Upon installation and loading the `cesR` package, we can use the `get_cescodes()` function to look up the code for the desired CES survey. In this case that is the 2019 CES phone survey.

```
# look up survey codes
get_cescodes()
```

In the printed data frame we can see that the argument code for the 2019 CES phone survey is 'ces2019_phone'. This code can be used with the `get_preview()` function to look up a truncated preview of the 2019 CES phone survey dataset.

```
# get preview of first 8 rows of 2019 CES phone survey dataset
get_preview("ces2019_phone", 8)
```

Using the same survey code argument with the `get_ces()` function we can retrieve the complete 2019 CES phone survey. Additionally, when `get_ces()` is called, the survey citation and link to repository prints to the console.

```
# call 2019 CES phone survey
get_ces("ces2019_phone")
```

To check that the dataset has loaded into RStudio correctly, it is best to check the dataset with the `head()` function from the `utils` package (R Core Team, 2020). Unfortunately, the `head()` function does not work with labelled data, so it is best to convert the values to factors using the `to_factor()` function from the `labelled` package (Larmarange, 2020).

```
# convert values to factor type
ces2019_phone <- to_factor(ces2019_phone)
head(ces2019_phone)
```

With the dataset loaded, the `select()` function from the `dplyr` package (Wickham et al., 2020a) is an efficient way of selecting the required variable columns. In this case, the language, phone type, weight, citizenship, year of birth, interviewer gender, identified gender, province, satisfaction with Canadian democracy, and most important election issue are selected using the column index. Additionally, the columns are not all named in an understandable way and so will be renamed using the `rename()` function from the `dplyr` package (Wickham et al., 2020a).

```
ces2019_phone_subset <- ces2019_phone %>%
  select(7, 8, 9, 20, 25:30)

head(ces2019_phone_subset)
```

We can see that some of the column names of the subset are not clearly named. To remedy this we can use the `rename()` function from the `dplyr` package (Wickham et al., 2020a). When using the `rename()` function, the order goes from new name to old name. To gain an idea of what to rename a column we can use the `get_question()` function to find the associated survey question.

```
get_question("ces2019_phone_subset", "q1")
get_question("ces2019_phone_subset", "q2")
get_question("ces2019_phone_subset", "q3")
get_question("ces2019_phone_subset", "q4")
get_question("ces2019_phone_subset", "q6")
get_question("ces2019_phone_subset", "q7")
```

Using the labels returned by the `get_question()` function we can better name the subset columns.

```
ces2019_phone_subset <- ces_phone_subset %>%
  rename(interviewer_gender = interviewer_gender_CES,
         language = language_CES,
         phonetype = phonetype_CES,
         weight = weight_CES,
         citizenship = q1,
         yob = q2,
         respondent_gender = q3,
         province_territory = q4,
         dmcrcy_satisfaction = q6,
         imprtnt_issue = q7)
```

This will have now created a subset of the CES 2019 phone survey with renamed variables and factor values.

As a final note, it is recommended to download the codebook for a requested survey dataset. These are available from the link printed on a survey call as well as on the package GitHub repository README.

4 Next steps and cautions

The `cesR` package is dependent upon functions from the `haven` (Wickham and Miller, 2020) and `labelled` (Larmarange, 2020) packages being able to read the CES survey datasets. Thus, changes to the `haven` or `labelled` packages may affect the functionality of the `cesR` package. As such, a future consideration would be to move away from this dependency to

make the `cesR` package more independent and self-contained.

Regarding the CES survey datasets, currently the datasets are downloaded from an associated GitHub repository. This means that a dataset will be downloaded as it currently exists on that repository. While these datasets are relatively stable (meaning the data is unchanging), updates are performed on the datasets to fix incorrect values or mislabelled variables from time to time. While the `cesR` package will be maintained to ensure the most up-to-date survey datasets are included, it may be the case that an update is not immediately performed on a dataset A step to eliminate this possible issue is to link the `get_ces()` call directly to the download URL of the survey as opposed to the current call to the GitHub repository.

The `cesR` package was written to not include datasets so as to minimize its size and speed up install times. However, as the package does need to download files, the speed at which the package functions is dependent upon the speed of a user's internet. This may slow down performance in some cases of poor internet connection.

Lastly, the CES surveys are generally divided into themed sections. A future step will be to build more subsets of the surveys, including the division of survey sections into their own subsets, so that topics may be more easily analysed.

References

Belanger, E. and Nadeau, R. (2005). Political trust and the vote in multiparty elections: The Canadian case. *European Journal of Political Research*, 44(1):121–146.

Bittner, A. (2018). Leaders always mattered: The persistence of personality in Canadian elections. *Electoral Studies*, 54:297–302.

Canadian Election Study (2019). *Welcome to the 2019 Canadian Election Study*. http://www.ces-eec.ca/.

Friendly, M., Dalzell, C., Monkman, M., and Murphy, D. (2020). *Lahman: Sean 'Lahman' Baseball Database*. R package version 8.0-0.

Gelfand, S. (2020). *opendatatoronto: Access the City of Toronto Open Data Portal*. R package version 0.1.3.

Gidengil, E., Blais, A., Nevitte, N., and Nadeau, R. (2001). The correlates and consequences of anti-partyism in the 1997 Canadian election. *Party Politics*, 7(4):491–513.

Larmarange, J. (2020). *labelled: Manipulating Labelled Data*. R package version 2.5.0.

Leeper, T. J. (2017). *dataverse: R Client for Dataverse 4*. R package version 0.2.0.

ODESI (2020). *ODESI*. https://search1.odesi.ca/#/.

R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

RStudio Team (2020). *RStudio: Integrated Development Environment for R*. RStudio, PBC, Boston, MA.

UBC (2015). *Canadian Election Study / Étude électorale canadienne*. https://ces-eec.arts.ubc.ca/english-section/surveys/.

von Bergmann, J., Shkolnik, D., and Jacobs, A. (2020). *cancensus: R package to access, retrieve, and work with Canadian Census data and geography*. R package version 0.3.2.

Wickham, H. (2014). *nasaweather: Collection of datasets from the ASA 2006 data expo*. R package version 0.1.

Wickham, H. (2020). *fueleconomy: EPA Fuel Economy Data*. R package version 1.0.0.

Wickham, H., François, R., Henry, L., and Müller, K. (2020a). *dplyr: A Grammar of Data Manipulation*. R package version 1.0.0.

Wickham, H. and Henry, L. (2020). *tidyr: Tidy Messy Data*. R package version 1.1.0.

Wickham, H., Hester, J., and Chang, W. (2020b). *devtools: Tools to Make Developing R Packages Easier*. R package version 2.3.1.

Wickham, H. and Miller, E. (2020). *haven: Import and Export 'SPSS', 'Stata' and 'SAS' Files*. R package version 2.3.1.

Wilkins-Laflamme, S. (2018). Islamophobia in Canada: Measuring the realities of negative attitudes toward Muslims and religious discrimination. *Canadian Review of Sociology/Revue canadienne de sociologie*, 55(1):86–110.