

Getting Organized to Do Data Science

Christopher Skovron & Jon Atwell
Northwestern University

June 20, 2018

From last time: getting help

- StackOverflow is a great resource
- May have to adjudicate between responses
- May have to adapt answers to your specific case

Use built-in help

```
?lm  
?read.csv  
?dplyr::rename
```

Built-in help will

- Tell you how functions work, to varying degrees of helpfulness
- Often provide some examples of how functions work
- Often still leave you clueless and needing to Google around some more

Not-so-useful help

grep {base}

R Documentation

Pattern Matching and Replacement

Description

`grep`, `grepl`, `regexpr`, `gregexpr` and `regexec` search for matches to argument `pattern` within each element of a character vector: they differ in the format of and amount of detail in the results.

`sub` and `gsub` perform replacement of the first and all matches respectively.

Usage

```
grep(pattern, x, ignore.case = FALSE, perl = FALSE, value = FALSE,  
      fixed = FALSE, useBytes = FALSE, invert = FALSE)
```

```
grepl(pattern, x, ignore.case = FALSE, perl = FALSE,  
       fixed = FALSE, useBytes = FALSE)
```

```
sub(pattern, replacement, x, ignore.case = FALSE, perl = FALSE,  
     fixed = FALSE, useBytes = FALSE)
```

```
gsub(pattern, replacement, x, ignore.case = FALSE, perl = FALSE,  
     fixed = FALSE, useBytes = FALSE)
```

```
regexpr(pattern, text, ignore.case = FALSE, perl = FALSE,  
        fixed = FALSE, useBytes = FALSE)
```

```
gregexpr(pattern, text, ignore.case = FALSE, perl = FALSE,  
         fixed = FALSE, useBytes = FALSE)
```

```
regexec(pattern, text, ignore.case = FALSE, perl = FALSE,
```

Not-so-useful help

Arguments

| | |
|--------------------------|--|
| <code>pattern</code> | character string containing a <u>regular expression</u> (or character string for <code>fixed = TRUE</code>) to be matched in the given character vector. Coerced by <code>as.character</code> to a character string if possible. If a character vector of length 2 or more is supplied, the first element is used with a warning. Missing values are allowed except for <code>regexpr</code> and <code>gregexpr</code> . |
| <code>x, text</code> | a character vector where matches are sought, or an object which can be coerced by <code>as.character</code> to a character vector. <u>Long vectors</u> are supported. |
| <code>ignore.case</code> | if <code>FALSE</code> , the pattern matching is <i>case sensitive</i> and if <code>TRUE</code> , case is ignored during matching. |
| <code>perl</code> | logical. Should Perl-compatible regexps be used? |
| <code>value</code> | if <code>FALSE</code> , a vector containing the (integer) indices of the matches determined by <code>grep</code> is returned, and if <code>TRUE</code> , a vector containing the matching elements themselves is returned. |
| <code>fixed</code> | logical. If <code>TRUE</code> , <code>pattern</code> is a string to be matched as is. Overrides all conflicting arguments. |
| <code>useBytes</code> | logical. If <code>TRUE</code> the matching is done byte-by-byte rather than character-by-character. See 'Details'. |
| <code>invert</code> | logical. If <code>TRUE</code> return indices or values for elements that do <i>not</i> match. |
| <code>replacement</code> | a replacement for matched pattern in <code>sub</code> and <code>gsub</code> . Coerced to character if possible. For <code>fixed = FALSE</code> this can include backreferences <code>"\1"</code> to <code>"\9"</code> to parenthesized subexpressions of <code>pattern</code> . For <code>perl = TRUE</code> only, it can also contain <code>"\U"</code> or <code>"\L"</code> to convert the rest of the replacement to upper or lower case and <code>"\E"</code> to end case conversion. If a character vector of length 2 or more is supplied, the first element is used with a warning. If <code>NA</code> , all elements in the result corresponding to matches will be set to <code>NA</code> . |

Some builtin help is more useful

R: Estimating Causal Effects in Conjoint Experiments ▾ Find in Topic

amce {cjoint} R Documentation

Estimating Causal Effects in Conjoint Experiments

Description

This function takes a dataset and a conjoint design and returns Average Marginal Component Effects (AMCEs) and Average Component Interaction Effects (ACIE) for the attributes specified in the formula. By default, this function assumes uniform randomization of attribute levels and no profile restrictions. If your design incorporates weighted randomization or restrictions on displayable profiles, first generate a design object using `makeDesign`. Interactions with respondent-level characteristics are handled by identifying relevant variables as respondent-varying.

Usage

```
amce(formula, data, design = "uniform",  
      respondent.varying = NULL, subset = NULL,  
      respondent.id = NULL, cluster = TRUE, na.ignore=FALSE,  
      weights = NULL, baselines = NULL)
```

Arguments

| | |
|----------------|--|
| formula | A <code>formula</code> object specifying the name of the outcome variable on the left-hand side and the attributes for which effects are to be estimated on the right-hand side. RHS attributes should be separated by + signs. Interaction effects can be specified using standard interaction syntax - joining attribute names using either : or *. However using the : syntax will produce the same results as * since missing base terms are automatically added to the formula. For example $Y \sim X1 + X2$ will return AMCEs for $X1$ and $X2$. $Y \sim X1 + X2 + X1:X2$ will return AMCEs for $X1$ and $X2$ along with an ACIE for $X1/X2$. $Y \sim X1*X2$ and $Y \sim X1:X2$ will produce identical results to $Y \sim X1 + X2 + X1:X2$. Note that you can place backticks around a variable name containing spaces in order to have <code>formula</code> interpret it as a single variable name. Any respondent characteristics must be designated as such in <code>redpondent.varying</code> . |
|----------------|--|

Some builtin help is more useful

Arguments

| | |
|---------------------------------|--|
| <code>formula</code> | A <code>formula</code> object specifying the name of the outcome variable on the left-hand side and the attributes for which effects are to be estimated on the right-hand side. RHS attributes should be separated by + signs. Interaction effects can be specified using standard interaction syntax - joining attribute names using either : or *. However using the : syntax will produce the same results as * since missing base terms are automatically added to the formula. For example $Y \sim X1 + X2$ will return AMCEs for X1 and X2. $Y \sim X1 + X2 + X1:X2$ will return AMCEs for X1 and X2 along with an ACIE for X1/X2. $Y \sim X1*X2$ and $Y \sim X1:X2$ will produce identical results to $Y \sim X1 + X2 + X1:X2$. Note that you can place backticks around a variable name containing spaces in order to have <code>formula</code> interpret it as a single variable name. Any respondent characteristics must be designated as such in <code>respondent.varying</code> . |
| <code>data</code> | A dataframe containing the outcome variable, attributes, respondent identifiers, respondent covariate data and sampling weights from a conjoint experiment. |
| <code>design</code> | Either the character string "uniform" or a <code>conjointDesign</code> object created by the <code>makeDesign</code> function. If a <code>conjointDesign</code> is not passed, the function will assume all attribute levels have an equal probability of being presented to a respondent and that no profiles are restricted. Defaults to "uniform". |
| <code>respondent.varying</code> | A vector of character strings giving the names of any respondent-varying characteristics being interacted with AMCEs or ACIEs in the <code>formula</code> . |
| <code>subset</code> | A logical vector with length <code>nrow(data)</code> denoting which rows in <code>data</code> should be included in estimation. This can for example be used to subset the data along respondent-level covariates. Defaults to <code>NULL</code> . |
| <code>respondent.id</code> | A character string indicating the column of <code>data</code> containing a unique identifier for each respondent. Defaults to <code>NULL</code> . |
| <code>cluster</code> | A logical indicating whether estimated standard errors should be clustered on <code>respondent.id</code> . Defaults to <code>TRUE</code> . |
| <code>na.ignore</code> | A logical indicating whether the function should ignore missing rows in <code>data</code> . If <code>FALSE</code> , <code>amce()</code> will raise an error if there are rows with missing values. Defaults to <code>FALSE</code> . |
| <code>weights</code> | A character string giving the name of the column in the <code>data</code> containing any survey weights. See documentation for <code>survey</code> package for more information. |

Some builtin help is more useful

Value

An object of class "amce" containing:

| | |
|---------------------------|--|
| attributes | A list containing the names of attributes. |
| baselines | Baseline levels for each attribute in <code>estimates</code> . Baselines determined using the first element of <code>levels()</code> . If a different baseline level is desired for an attribute, use the <code>relevel()</code> function on the variable prior to calling the <code>amce()</code> routine or supply an alternative baseline in <code>baselines</code> argument. |
| continuous | List of quantiles for any non-factor variables, whether attributes or respondent varying. |
| data | The original data. |
| estimates | A list containing AMCE and ACIE estimates for each attribute in <code>formula</code> . Each element of <code>estimates</code> corresponds to a single attribute or interaction. |
| formula | The formula passed to the <code>amce()</code> routine. |
| samplesize_prof | The number of valid profiles (rows) in the dataset |
| user.names | A vector with the original user supplied names for any attributes. These may differ from the attribute names in <code>estimates</code> if the original names contain spaces. |
| vcov_prof | The modified variance-covariance matrix for AMCE and ACIE estimates. Incorporates cluster corrections as well as attribute dependencies. Profile varying attributes only. |
| numrespondents | The number of respondents in the dataset (if <code>respondent.id</code> is not NULL). |
| respondent.varying | Names of respondent-varying variables, if any. |
| cond.formula | The formula used for calculating estimates conditional on respondent varying characteristics. Only returned when respondent-varying characteristics are present. |
| cond.estimates | A list containing estimated effects of respondent-varying characteristics conditional on attribute values. Each element of <code>cond.estimates</code> corresponds to a single attribute or interaction. Only returned when respondent-varying characteristics are present. To obtain AMCE and ACIE estimates conditional on the values of the |

Some builtin help is more useful

See Also

`summary.amce` for summaries and `plot.amce` for generating a coefficient plot using `ggplot2`.

`makeDesign` to create `conjointDesign` objects.

Examples

```
# Immigration Choice Conjoint Experiment Data from Hainmueller et. al. (2014).
data("immigrationconjoint")
data("immigrationdesign")

# Run AMCE estimator using all attributes in the design
results <- amce(Chosen_Immigrant ~ Gender + Education + `Language Skills` +
  `Country of Origin` + Job + `Job Experience` + `Job Plans` +
  `Reason for Application` + `Prior Entry`, data=immigrationconjoint,
  cluster=TRUE, respondent.id="CaseID", design=immigrationdesign)

# Print summary
summary(results)

## Not run:
# Run AMCE estimator using all attributes in the design with interactions
interaction_results <- amce(Chosen_Immigrant ~ Gender + Education + `Language Skills` +
  `Country of Origin` + Job + `Job Experience` + `Job Plans` +
  `Reason for Application` + `Prior Entry` + Education:`Language Skills` +
  Job: `Job Experience` + `Job Plans`:`Reason for Application`,
  data=immigrationconjoint, cluster=TRUE, respondent.id="CaseID",
  design=immigrationdesign)

# Print summary
summary(interaction_results)

# create weights in data
weights <- runif(nrow(immigrationconjoint))
```

Asking for help

- You will want a “minimal working example”
- [Here's a good guide for writing one in R from Jared Knowles](#)
- The most efficient way to ask for help, in exponentially decreasing order: Ask Google, ask people you know, ask strangers online

Get organized

Your audiences, in decreasing order

- Your future self
- Your collaborators
- Replicators

Good organizational principles

One script per task

- In general, each script should do a specific high-level task
- Figuring out where to break up scripts is more of an art than a science, but some rules of thumb
- If you are at a point in your workflow where it would make sense to save something - a dataset, a model object, a few figures, etc, that is often a good time for a new script

Mind the order

- Complex tasks will require multiple scripts
- Use organizing tools like RStudio's Projects
- Number scripts in order: `0_clean_data.R`,
`1_merge_census_data.R`,
`3_fit_multilevel_models.R`

Automate everything that can be automated

- Saves yourself time
- Prevents mistakes
- Sometimes comes at the cost of more time up front and code readability

DON'T REPEAT YOURSELF

- Inefficient
- Invites mistakes
- Makes even simple tweaks to code take forever to implement
- Ask yourself, “could this code do this task 100 times?”

Don't copy and paste

- Call things from where they are stored
- Typically, this will be on your disk or on some remote/cloud disk you can access from your machine
- If your code calls for the same task to be repeated, find ways to automate

Set up lists you can iterate over

- If you're doing the same task over and over, you want to iterate over a list
- Strategies for doing this depend on task and language
- for loops
- apply functions

Understand your file structure

- Find an organization that works for you, and be consistent
- One approach to file structures for R projects
- An excellent tutorial about organized workflow in R

Paths

- Paths are how your program finds what it needs to
- You shouldn't be clicking on datasets or files to load them

Modern IDEs can autocomplete paths

```
read.csv(file, header = TRUE, sep = ",", quote = "\"", dec = ".", fill =  
TRUE, comment.char = "#", ...)  
```{r}  
data = read.csv()
```
```

```
```{r}  
data = read.csv("~/Dr|")
```
```

Dropbox ~

```
```{r}  
data = read.csv("~/Dropbox/Data/")
```
```

Setting a word
- Each script should
- This is where the
script, etc.
- R:

| | |
|----------------------------------|------------------|
| 2016 presvote | .../Dropbox/Data |
| 2016-CCES-MRP | .../Dropbox/Data |
| ACS 2015 CDs | .../Dropbox/Data |
| ACS 2015 SLDs | .../Dropbox/Data |
| ACS 2016 Congressional Districts | .../Dropbox/Data |
| ACS 2016 SLDs | .../Dropbox/Data |
| ACS 2016 states | .../Dropbox/Data |

any files you create using the

```
```{r}  
data = read.csv("~/Dropbox/Data/CCES 2016/cces16_raw_lower.csv")
```
```

Setting a working directory

- Each script should have a “working directory”
- This is where your program will search for files referenced in the script, write any files you create using the script, etc.
- R:

```
setwd("/Users/cskovron/Dropbox/Research/ncs-constituent-eval/analysis")  
# on Mac is equivalent to  
setwd("~/Dropbox/Research/ncs-constituent-eval/analysis")
```

- Python:

```
import os  
path="/Users/cskovron/Dropbox/Research/ncs-constituent-eval/analysis"  
os.chdir(path)
```

- Command line:

```
cd "~/Dropbox/Research/ncs-constituent-eval/analysis"
```


In R, convention is now to use RStudio “Projects”

- This improves reproducibility but is a little more advanced, so I encourage you to look at it on your own if you are an R specialist

Navigating, relational paths

- After you set your working directory, you'll want to use it to navigate around
- Typically in your program, you can refer to anything in the working directory without prefacing it with anything

```
dat <- read.csv("some-file-in-your-working-directory.csv", stringsAsFactor)
```

Setting relative paths - Unix-based systems including Mac

- You can set paths at levels outside your working directory
- ~ Your home directory (on my Mac, `/Users/cskovron/`)
- . The current directory (`./images/` is a subfolder of the working directory called `images`)
- .. The parent of the current directory (the directory the working directory is in)

Don't change your working directory to save to a subfolder

- Just do:

```
write.csv(some.data.to.save, "./data-subfolder/data-filename.csv")
```

Paths don't play nicely between Windows and Unix systems

- A tutorial on making your Python code portable between Windows and Unix

Version control

- Simplest definition – lets you revert to the past
- Track changes made by collaborators
- Document your contributions to code

Version control options

- Git/github
 - [A tutorial on getting started with GitHub Desktop](#)
 - [A tutorial on GitHub integration with R Studio](#)
- Dropbox is a good option, though less control than git
- Collaboration is tough - live editing code for Python and R are coming but not common right now. More likely, one collaborator should be in a file at a time

Tidy data principles

- Every data task will be different, but tidy data principles will help you keep organized
- Much of your workflow will be getting data inputs into a tidy format
- From the beginning, lay out what you need your dataset to look like to do the analyses you want, then work backward from there
- Hadley's vignette on tidy data principles
- Chapter of R for Data Science on tidy data

Tidy data has

- Each variable forms a column.
- Each observation forms a row.
- Each type of observational unit forms a table.

Tidy data does not have

- Column headers are values, not variable names.
- Multiple variables are stored in one column.
- Variables are stored in both rows and columns.
- Multiple types of observational units are stored in the same table.
- A single observational unit is stored in multiple tables.

Long vs wide data

dcast formula `dcast(aql, month + day ~ variable, value.var = "value")`

| | | |
|--|---|-----------------------|
| ID variables (left side of formula) | Variable to swing into column names (right side of formula) | Values (value.var) |
|--|---|-----------------------|

Long-format data

| month | day | variable | value |
|-------|-----|----------|-------|
| 5 | 1 | ozone | 41 |
| 5 | 2 | ozone | 36 |
| 5 | 3 | ozone | 12 |
| 5 | 4 | ozone | 18 |
| 5 | 5 | ozone | NA |
| 5 | 6 | ozone | 28 |

Wide-format data

| month | day | ozone | solar.r | wind | temp |
|-------|-----|-------|---------|------|------|
| 5 | 1 | 41 | 190 | 7.4 | 67 |
| 5 | 2 | 36 | 118 | 8.0 | 72 |
| 5 | 3 | 12 | 149 | 12.6 | 74 |
| 5 | 4 | 18 | 313 | 11.5 | 62 |
| 5 | 5 | NA | NA | 14.3 | 56 |
| 5 | 6 | 28 | NA | 14.9 | 66 |

Relational data

- Some data has structures more complex than simple tables
- For example, Netflix has a database where each user has a table of movies they've watched and a separate table for each movie of the users who have watched it
- This is “relational data”
- It's often, but not always, big
- Requires special tools, usually SQL

Checking up on your data cleaning

- glance at your data:
 - `View()` (but be careful!)
 - `summary()` (be careful on big datasets)
 - `head()` and `tail()`
 - `tibble::glimpse()`
 - `is.na()` and `sum(is.na())`

Set yourself up to do things iteratively

- In one paper, I do the same analysis for many survey items
- To automate this, I made a helper file called `issues.names.titles.csv`

| | issue | question.text | issue.short | yes.is.liberal | econ.issue |
|----|---------------|--|---|----------------|------------|
| 1 | nateconbetter | Over the past year the nation's economy has got... | National economy improving | NA | TRUE |
| 2 | raisefueleff | Raise required fuel efficiency for the average automo... | Raise fuel efficiency | TRUE | FALSE |
| 3 | raiseminrenew | Require a minimum amount of renewable fuels in the ... | Require renewable energy | TRUE | FALSE |
| 4 | elimmandsent | Eliminate mandatory minimum sentences for non-viol... | Eliminate mandatory minimums for drug offenders | TRUE | FALSE |
| 5 | incprisonsent | Increase prison sentences for felons who have already... | Three strikes for felons | FALSE | FALSE |
| 6 | raiseminwage | Raise the federal minimum wage to \$12 an hour by 2... | Raise the minimum wage | TRUE | TRUE |
| 7 | whitepladv | White people in the U.S. have certain advantages beca... | White people have advantages | TRUE | FALSE |
| 8 | banassault | On the issue of gun regulation, I support banning ass... | Ban assault rifles | TRUE | FALSE |
| 9 | legalstatilim | Grant legal status to all illegal immigrants who have h... | Grant legal status to unauthorized immigrants | TRUE | FALSE |
| 10 | abortionilleg | Make abortions illegal in all circumstances. | Make abortion illegal | FALSE | FALSE |
| 11 | incspend_ed | The state legislature should increase spending on ed... | Increase education spending | TRUE | TRUE |
| 12 | incspend_hc | The state legislature should increase spending on hea... | Increase health care spending | TRUE | TRUE |
| 13 | incspend_law | The state legislature should increase spending on law... | Increase law enforcement spending | NA | TRUE |
| 14 | incspend_inf | The state legislature should increase spending on infr... | Increase infrastructure spending | TRUE | TRUE |
| 15 | impt_ab | The issue of abortion is of very high importance to me. | Importance: abortion | NA | FALSE |
| 16 | impt_env | The issue of the environment is of very high importan... | Importance: environment | NA | FALSE |
| 17 | impt_ssm | The issue of gay marriage is of very high importance ... | Importance: same sex marriage | NA | FALSE |
| 18 | impt_gunc | The issue of gun control is of very high importance to... | Importance: gun control | NA | FALSE |
| 19 | impt_hc | The issue of health care is of very high importance to ... | Importance: health care | NA | TRUE |
| 20 | impt_imm | The issue of immigration is of very high importance t... | Importance: immigration | NA | FALSE |
| 21 | impt_jobs | The issue of jobs is of very high importance to me. | Importance: jobs | NA | TRUE |
| 22 | impt_racer | The issue of race relations is of very high importance ... | Importance: race relations | NA | FALSE |
| 23 | impt_taxes | The issue of taxes is of very high importance to me. | Importance: taxes | NA | TRUE |

Tricks for doing things iteratively

- Use `paste ()` and `paste0 ()` to help write captions and labels
- Can select columns using variables: `data[, issue]`, if `issue` is a character vector, selects just that column. Loop over issues

Where to work? Development environments

- Let your software do some of the work for you
- RStudio projects
- RStudio gets new features every day that help you stay organized
- RMarkdown allows you to integrate R code and writing to produce reports in HTML and PDF, slides, etc. Very flexible and extendable

Get organized to learn in the future

- Follow #rstats on twitter
- [I made a twitter list of good R follows here](#)
- Star your favorite packages on github and follow developers