Day 11: R Packages and Quanteda

Erin Rossiter

17 April, 2023

(Class pre-requisites for Windows users)

1. Download the appropriate version of RTools for your version of R: https://cran.r-project.org/bin/windows/Rtools/
2. Install RTools by running the downloaded executable
3. Set the path to RTools in your system environment variables. To do this, right-click on "My Computer" > "Properties" > "Advanced system settings" > "Environment Variables".

Under "System Variables", find the "PATH" variable and click "Edit". Add the path to the RTools bin folder (e.g., `C:\Rtools\bin`) to the list of paths.

Example: `C:\\Program Files\\R\\R-4.2.2\\bin; C:\\Rtools;}`

4. Restart your R session to ensure that the changes to the system environment variables take effect.

Announcements

Announcements

- PS09 due tomorrow
- Anything else?

Announcements

- PS09 due tomorrow
- Anything else?

Roadmap

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
  - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
  - Base R
  - ggplot
  - hopefully more fun stuff if time. . .

Roadmap

Last unit:

- data cleaning and wrangling

Final unit:

- advanced topics

Today:

- R packages
- Basic text-as-data (Quanteda R package)
    - Not any methods, just working in R
- Rcpp

Next time:

- Data viz
    - Base R
    - ggplot
    - hopefully more fun stuff if time. . .

R Packages

What is an R package?

- a collection of (mainly) functions, but also data sets, help files, etc.
- a way to share code and data with other R users
  - remember R is open-source language
  - large and active community of R users **and developers** who create R packages
  - R packages are why it is so impactful to learn R vs. another language
  - usually, new stats methods in political science will make their way to R first

What is an R package?

- a collection of (mainly) functions, but also data sets, help files, etc.
- a way to share code and data with other R users
  - remember R is open-source language
  - large and active community of R users **and developers** who create R packages
  - R packages are why it is so impactful to learn R vs. another language
  - usually, new stats methods in political science will make their way to R first

What is an R package?

- a collection of (mainly) functions, but also data sets, help files, etc.
- a way to share code and data with other R users
  - remember R is open-source language
  - large and active community of R users **and developers** who create R packages
  - R packages are why it is so impactful to learn R vs. another language
  - usually, new stats methods in political science will make their way to R first

What is an R package?

- a collection of (mainly) functions, but also data sets, help files, etc.
- a way to share code and data with other R users
  - remember R is open-source language
  - large and active community of R users **and developers** who create R packages
  - R packages are why it is so impactful to learn R vs. another language
  - usually, new stats methods in political science will make their way to R first

What is an R package?

- a collection of (mainly) functions, but also data sets, help files, etc.
- a way to share code and data with other R users
  - remember R is open-source language
  - large and active community of R users **and developers** who create R packages
  - R packages are why it is so impactful to learn R vs. another language
  - usually, new stats methods in political science will make their way to R first

What is an R package?

- a collection of (mainly) functions, but also data sets, help files, etc.
- a way to share code and data with other R users
  - remember R is open-source language
  - large and active community of R users **and developers** who create R packages
  - R packages are why it is so impactful to learn R vs. another language
  - usually, new stats methods in political science will make their way to R first

Why make a package?

- You have a novel statistical method
- You want credit for something, perhaps citations even (see this paper)
- “Simple” way to share complex code/results/data.
- Coherent way to organize:
  - Data
  - Documentation/explanation
  - Meta-data
  - Execute complex code
- Free Hadley Wickham book if you want to make your own

Why make a package?

- You have a novel statistical method
- You want credit for something, perhaps citations even (see this paper)
- "Simple" way to share complex code/results/data.
- Coherent way to organize:
  - Data
  - Documentation/explanation
  - Meta-data
  - Execute complex code
- Free Hadley Wickham book if you want to make your own

Why make a package?

- You have a novel statistical method
- You want credit for something, perhaps citations even (see this paper)
- “Simple” way to share complex code/results/data.
- Coherent way to organize:
    - Data
    - Documentation/explanation
    - Meta-data
    - Execute complex code
- Free Hadley Wickham book if you want to make your own

Why make a package?

- You have a novel statistical method
- You want credit for something, perhaps citations even (see this paper)
- “Simple” way to share complex code/results/data.
- Coherent way to organize:
  - Data
  - Documentation/explanation
  - Meta-data
  - Execute complex code
- Free Hadley Wickham book if you want to make your own

Why make a package?

- You have a novel statistical method
- You want credit for something, perhaps citations even (see this paper)
- “Simple” way to share complex code/results/data.
- Coherent way to organize:
  - Data
  - Documentation/explanation
  - Meta-data
  - Execute complex code
- Free Hadley Wickham book if you want to make your own

Where to find/publish R packages

Where to find/publish R packages

- CRAN
- Github

CRAN

CRAN is the Comprehensive R Archive Network

- As of today, there are 19365 available packages! See here
- List by name

CRAN

CRAN is the Comprehensive R Archive Network

- As of today, there are 19365 available packages! See here
- List by name

CRAN

CRAN is the Comprehensive R Archive Network

- As of today, there are 19365 available packages! See here
- List by name

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

- CRAN is the primary repository for R packages
- But, some developers prefer to distribute their packages through GitHub
- Easier, quicker, less rules and hassle!
  - Ex: Lots of dependencies that CRAN won't allow
- Or even kicked off CRAN...

2. Maybe they are still in development

- Version control
- Collaboration
- Distribution
- Documentation
- ... as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

- CRAN is the primary repository for R packages
- But, some developers prefer to distribute their packages through GitHub
- Easier, quicker, less rules and hassle!
  - » Ex: Lots of dependencies that CRAN won't allow
- Or even kicked off CRAN. . .

2. Maybe they are still in development

- Version control
- Collaboration
- Distribution
- Documentation
- . . . as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

   - CRAN is the primary repository for R packages
   - But, some developers prefer to distribute their packages through GitHub
   - Easier, quicker, less rules and hassle!
     - Ex: Lots of dependencies that CRAN won't allow
   - Or even kicked off CRAN...

2. Maybe they are still in development

   - Version control
   - Collaboration
   - Distribution
   - Documentation
   - ... as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

   - CRAN is the primary repository for R packages
   - But, some developers prefer to distribute their packages through GitHub
   - Easier, quicker, less rules and hassle!
     - Ex: Lots of dependencies that CRAN won't allow
   - Or even kicked off CRAN...

2. Maybe they are still in development

   - Version control
   - Collaboration
   - Distribution
   - Documentation
   - ... as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

- CRAN is the primary repository for R packages
- But, some developers prefer to distribute their packages through GitHub
- Easier, quicker, less rules and hassle!
  - Ex: Lots of dependencies that CRAN won't allow
- Or even kicked off CRAN. . .

2. Maybe they are still in development

- Version control
- Collaboration
- Distribution
- Documentation
- . . . as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

- CRAN is the primary repository for R packages
- But, some developers prefer to distribute their packages through GitHub
- Easier, quicker, less rules and hassle!
  - » Ex: Lots of dependencies that CRAN won't allow
- Or even kicked off CRAN...

2. Maybe they are still in development

- Version control
- Collaboration
- Distribution
- Documentation
- ... as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

- CRAN is the primary repository for R packages
- But, some developers prefer to distribute their packages through GitHub
- Easier, quicker, less rules and hassle!
  - Ex: Lots of dependencies that CRAN won't allow
- Or even kicked off CRAN…

2. Maybe they are still in development

- Version control
- Collaboration
- Distribution
- Documentation
- … as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

  - CRAN is the primary repository for R packages
  - But, some developers prefer to distribute their packages through GitHub
  - Easier, quicker, less rules and hassle!
    - Ex: Lots of dependencies that CRAN won't allow
  - Or even kicked off CRAN...

2. Maybe they are still in development

  - Version control
  - Collaboration
  - Distribution
  - Documentation
  - ... as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

- CRAN is the primary repository for R packages
- But, some developers prefer to distribute their packages through GitHub
- Easier, quicker, less rules and hassle!
  - » Ex: Lots of dependencies that CRAN won't allow
- Or even kicked off CRAN. . .

2. Maybe they are still in development

- Version control
- Collaboration
- Distribution
- Documentation
- . . . as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

- CRAN is the primary repository for R packages
- But, some developers prefer to distribute their packages through GitHub
- Easier, quicker, less rules and hassle!
  - Ex: Lots of dependencies that CRAN won't allow
- Or even kicked off CRAN…

2. Maybe they are still in development

- Version control
- Collaboration
- Distribution
- Documentation
- … as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

- CRAN is the primary repository for R packages
- But, some developers prefer to distribute their packages through GitHub
- Easier, quicker, less rules and hassle!
  - » Ex: Lots of dependencies that CRAN won't allow
- Or even kicked off CRAN…

2. Maybe they are still in development

- Version control
- Collaboration
- Distribution
- Documentation
- … as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

- CRAN is the primary repository for R packages
- But, some developers prefer to distribute their packages through GitHub
- Easier, quicker, less rules and hassle!
    - Ex: Lots of dependencies that CRAN won't allow
- Or even kicked off CRAN...

2. Maybe they are still in development

- Version control
- Collaboration
- Distribution
- Documentation
- ... as we know, all great on Github

GitHub

You might find R packages on Github. Why?

1. Maybe this is their final home

- CRAN is the primary repository for R packages
- But, some developers prefer to distribute their packages through GitHub
- Easier, quicker, less rules and hassle!
    - » Ex: Lots of dependencies that CRAN won't allow
- Or even kicked off CRAN...

2. Maybe they are still in development

- Version control
- Collaboration
- Distribution
- Documentation
- ... as we know, all great on Github

Developing an R Package

Package Structure

An R package *must* contain:

- R functions (.R files)
- Documentation (.Rd files)
- Meta-data (NAMESPACE and DESCRIPTION)

An R package *must* contain:

- R functions (.R files)
- Documentation (.Rd files)
- Meta-data (NAMESPACE and DESCRIPTION)

An R package *must* contain:

- R functions (.R files)
- Documentation (.Rd files)
- Meta-data (NAMESPACE and DESCRIPTION)

An R package *must* contain:

- R functions (.R files)
- Documentation (.Rd files)
- Meta-data (NAMESPACE and DESCRIPTION)

An R package *can* contain:

- Datasets
- Demo files
- Class structures (and helper functions)
- Compiled code
- README
- More

An R package *can* contain:

- Datasets
- Demo files
- Class structures (and helper functions)
- Compiled code
- README
- More

An R package *can* contain:

- Datasets
- Demo files
- Class structures (and helper functions)
- Compiled code
- README
- More

An R package *can* contain:

- Datasets
- Demo files
- Class structures (and helper functions)
- Compiled code
- README
- More

An R package *can* contain:

- Datasets
- Demo files
- Class structures (and helper functions)
- Compiled code
- README
- More

An R package *can* contain:

- Datasets
- Demo files
- Class structures (and helper functions)
- Compiled code
- README
- More

Example: squaresPack

What is the necessary file structure to create an R package with these advanced statistical methods? (on board)

```r
addSquares <- function(x, y){
  return(list(square = (x^2 + y^2), x = x, y = y))
}


subtractSquares <- function(x, y){
  return(list(square = (x^2 - y^2), x = x, y = y))
}
```

In more detail

- Directory must have the name of the package
- DESCRIPTION file must have **exactly** that name. This contains required meta-data about the package (e.g., version number).
- NAMESPACE determined what functions or objects will be available in the global environment and sets up a package-specific namespace.
- The R directory contains *.R files with your scripts/functions/etc.
- The man directory contains the *.Rd help files.

In more detail

- Directory must have the name of the package
- `DESCRIPTION` file must have **exactly** that name. This contains required meta-data about the package (e.g., version number).
- `NAMESPACE` determined what functions or objects will be available in the global environment and sets up a package-specific namespace.
- The `R` directory contains `*.R` files with your scripts/functions/etc.
- The `man` directory contains the `*.Rd` help files.

In more detail

- Directory must have the name of the package
- DESCRIPTION file must have **exactly** that name. This contains required meta-data about the package (e.g., version number).
- NAMESPACE determined what functions or objects will be available in the global environment and sets up a package-specific namespace.
- The R directory contains *.R files with your scripts/functions/etc.
- The man directory contains the *.Rd help files.

In more detail

- Directory must have the name of the package
- DESCRIPTION file must have **exactly** that name. This contains required meta-data about the package (e.g., version number).
- NAMESPACE determined what functions or objects will be available in the global environment and sets up a package-specific namespace.
- The R directory contains *.R files with your scripts/functions/etc.
- The man directory contains the *.Rd help files.

In more detail

- Directory must have the name of the package
- DESCRIPTION file must have **exactly** that name. This contains required meta-data about the package (e.g., version number).
- NAMESPACE determined what functions or objects will be available in the global environment and sets up a package-specific namespace.
- The R directory contains *.R files with your scripts/functions/etc.
- The man directory contains the *.Rd help files.

Folder: `R`

Put your `R` scripts in the `R` directory.

- The simplest convention is to include one `R` function in each file
- In many instances you will find that files contain multiple `R` functions, especially if there is some class system.

Folder: `R`

Put your `R` scripts in the `R` directory.

- The simplest convention is to include one `R` function in each file
- In many instances you will find that files contain multiple `R` functions, especially if there is some class system.

Folder: `R`

Put your `R` scripts in the `R` directory.

- The simplest convention is to include one `R` function in each file
- In many instances you will find that files contain multiple `R` functions, especially if there is some class system.

Folder: man

- These are your help files!
- Written in markup language called “rd” (R documentation)

```
\name{addSquares}
\alias{addSquares}
\title{Adding squared values}
\usage{
  addSquares(x, y)
}
\usage{
  addSquares(x, y)
}
\arguments{
 \item{x}{A numeric object}
 \item{y}{A numeric object with the same dimensionality as
  \code{x}.}
}
\value{
  A list with the elements
 \item{squares}{The sum of the  squared values}
 \item{x}{The first object input}
```

Folder: `man`

- These are your help files!
- Written in markup language called “rd” (R documentation)

```
\name{addSquares}
\alias{addSquares}
\title{Adding squared values}
\usage{
  addSquares(x, y)
}
\usage{
  addSquares(x, y)
}
\arguments{
 \item{x}{A numeric object}
 \item{y}{A numeric object with the same dimensionality as
  \code{x}.}
}
\value{
  A list with the elements
 \item{squares}{The sum of the  squared values}
 \item{x}{The first object input}
```

Folder: man

- These are your help files!
- Written in markup language called “rd” (R documentation)

```
\name{addSquares}
\alias{addSquares}
\title{Adding squared values}
\usage{
  addSquares(x, y)
}
\usage{
  addSquares(x, y)
}
\arguments{
 \item{x}{A numeric object}
 \item{y}{A numeric object with the same dimensionality as
  \code{x}.}
}
\value{
  A list with the elements
 \item{squares}{The sum of the  squared values}
 \item{x}{The first object input}
```

File: `DESCRIPTION`

Contains:

- package name
- formal title
- current version number
- the date for the version release
- the name and contact information of of the author and maintainer
- dependencies
- list of the files in the `R` subdirectory

File: DESCRIPTION

Contains:

- package name
- formal title
- current version number
- the date for the version release
- the name and contact information of of the author and maintainer
- dependencies
- list of the files in the R subdirectory

File: `DESCRIPTION`

Contains:

- package name
- formal title
- current version number
- the date for the version release
- the name and contact information of of the author and maintainer
- dependencies
- list of the files in the `R` subdirectory

File: `DESCRIPTION`

Contains:

- package name
- formal title
- current version number
- the date for the version release
- the name and contact information of of the author and maintainer
- dependencies
- list of the files in the `R` subdirectory

File: DESCRIPTION

Contains:

- package name
- formal title
- current version number
- the date for the version release
- the name and contact information of of the author and maintainer
- dependencies
- list of the files in the R subdirectory

File: `DESCRIPTION`

Contains:

- package name
- formal title
- current version number
- the date for the version release
- the name and contact information of of the author and maintainer
- dependencies
- list of the files in the `R` subdirectory

File: DESCRIPTION

Contains:

- package name
- formal title
- current version number
- the date for the version release
- the name and contact information of of the author and maintainer
- dependencies
- list of the files in the R subdirectory

File: `DESCRIPTION`

Contains:

- package name
- formal title
- current version number
- the date for the version release
- the name and contact information of of the author and maintainer
- dependencies
- list of the files in the `R` subdirectory

Example:

```
Package: squaresPack
Title: Adding and subtracting squared values
Version: 0.1
Author: Jacob M. Montgomery
Maintainer: Jacob M. Montgomery <jacob.montgomery@wustl.edu>
Description: Find sum and difference of squared values
Depends:
    R (>= 3.0.0)
License: GPL (>= 2)
Suggests:
    devtools
Collate:
   `addSquares.R'
   `subtractSquares.R'
```

File: NAMESPACE

This (can be) the most difficult part, and is aimed at setting up a package specific environment and controlling what functions users can see and/or use directly.

At a minimum, it needs to read in the functions, class definitions etc. that are “available” to R.

The contents of the NAMESPACE file for this package are:

```
export(addSquares)
export(subtractSquares)
```

File: NAMESPACE

This (can be) the most difficult part, and is aimed at setting up a package specific environment and controlling what functions users can see and/or use directly.

At a minimum, it needs to read in the functions, class definitions etc. that are "available" to R.

The contents of the NAMESPACE file for this package are:

```
export(addSquares)
export(subtractSquares)
```

File: NAMESPACE

This (can be) the most difficult part, and is aimed at setting up a package specific environment and controlling what functions users can see and/or use directly.

At a minimum, it needs to read in the functions, class definitions etc. that are “available” to R.

The contents of the NAMESPACE file for this package are:

```
export(addSquares)
export(subtractSquares)
```

File: NAMESPACE

This (can be) the most difficult part, and is aimed at setting up a package specific environment and controlling what functions users can see and/or use directly.

At a minimum, it needs to read in the functions, class definitions etc. that are “available” to R.

The contents of the NAMESPACE file for this package are:

```
export(addSquares)
export(subtractSquares)
```

Beyond scope for today

- Adding test files, see here
- Dependencies
- Adding data (its much like adding R functions)
- Final steps for submitting package to CRAN, meeting their rules, etc.
  - See Hadley Wickham book linked at beginning

Beyond scope for today

- Adding test files, see here
- Dependencies
- Adding data (its much like adding R functions)
- Final steps for submitting package to CRAN, meeting their rules, etc.
  - See Hadley Wickham book linked at beginning

Beyond scope for today

- Adding test files, see here
- Dependencies
- Adding data (its much like adding R functions)
- Final steps for submitting package to CRAN, meeting their rules, etc.
  - See Hadley Wickham book linked at beginning

Beyond scope for today

- Adding test files, see here
- Dependencies
- Adding data (its much like adding R functions)
- Final steps for submitting package to CRAN, meeting their rules, etc.
    - See Hadley Wickham book linked at beginning

Beyond scope for today

- Adding test files, see here
- Dependencies
- Adding data (its much like adding R functions)
- Final steps for submitting package to CRAN, meeting their rules, etc.
  - See Hadley Wickham book linked at beginning

Using R packages

Downloading

Downloading from CRAN

- We need to downloads and install the packages we want onto our local computers
- You'll ask R to download the package from a repository, probably CRAN or Github
  - That's why we don't have all 19000+ package already at our disposal!

```
install.packages("A3")
```

Downloading from CRAN

- We need to downloads and install the packages we want onto our local computers
- You'll ask R to download the package from a repository, probably CRAN or Github
  - That's why we don't have all 19000+ package already at our disposal!

```
install.packages("A3")
```

Downloading from CRAN

- We need to downloads and install the packages we want onto our local computers
- You'll ask R to download the package from a repository, probably CRAN or Github
    - » That's why we don't have all 19000+ package already at our disposal!

```
install.packages("A3")
```

Downloading from CRAN

- We need to downloads and install the packages we want onto our local computers
- You'll ask R to download the package from a repository, probably CRAN or Github
  - That's why we don't have all 19000+ package already at our disposal!

```
install.packages("A3")
```

Downloading from Github

- We use the `install_github()` in the `devtools` package (yes, this is meta)
- The format is: `install_github("username/repo")`

```r
# This package has functions that help install
# other packages on GitHub!
install.packages("devtools")

# Load it (next topic in slides)
library(devtools)

# An Example
install_github("erossiter/blockclustr")
```

Downloading from Github

- We use the `install_github()` in the `devtools` package (yes, this is meta)
- The format is: `install_github("username/repo")`

```r
# This package has functions that help install
# other packages on GitHub!
install.packages("devtools")

# Load it (next topic in slides)
library(devtools)

# An Example
install_github("erossiter/blockclustr")
```

Downloading from Github

- We use the `install_github()` in the `devtools` package (yes, this is meta)
- The format is: `install_github("username/repo")`

```
# This package has functions that help install
# other packages on GitHub!
install.packages("devtools")

# Load it (next topic in slides)
library(devtools)

# An Example
install_github("erossiter/blockclustr")
```

Loading

Loading the package for use

- We download it, so entire package is now on our computer
- Now, we want to be able to use the functions (or anything else) in it!
- Loading the package is the same whether downloaded from CRAN or GitHub
- `library()` load everything in the package into memory

```
library(A3)

## Loading required package: xtable

## Loading required package: pbapply

library(blockclustr)
```

Loading the package for use

- We download it, so entire package is now on our computer
- Now, we want to be able to use the functions (or anything else) in it!
- Loading the package is the same whether downloaded from CRAN or GitHub
- `library()` load everything in the package into memory

```
library(A3)
```

```
## Loading required package: xtable

## Loading required package: pbapply
```

```
library(blockclustr)
```

Loading the package for use

- We download it, so entire package is now on our computer
- Now, we want to be able to use the functions (or anything else) in it!
- Loading the package is the same whether downloaded from CRAN or GitHub
- `library()` load everything in the package into memory

```
library(A3)
```

```
## Loading required package: xtable
```

```
## Loading required package: pbapply
```

```
library(blockclustr)
```

Loading the package for use

- We download it, so entire package is now on our computer
- Now, we want to be able to use the functions (or anything else) in it!
- Loading the package is the same whether downloaded from CRAN or GitHub
- `library()` load everything in the package into memory

```
library(A3)
```

```
## Loading required package: xtable
```

```
## Loading required package: pbapply
```

```
library(blockclustr)
```

Loading the package for use

- We download it, so entire package is now on our computer
- Now, we want to be able to use the functions (or anything else) in it!
- Loading the package is the same whether downloaded from CRAN or GitHub
- `library()` load everything in the package into memory

```
library(A3)
```

```
## Loading required package: xtable

## Loading required package: pbapply
```

```
library(blockclustr)
```

Or access without loading

- Syntax is ‘`package::function()`
- Grabs function from the package
- Useful:
  - to be very clear where function comes from
  - if same function name in different packages
- Examples:

```
blockclustr::blockclustr()
dplyr::left_join()
rvest::read_html()
```

Or access without loading

- Syntax is ‘package::function()
- Grabs function from the package
- Useful:
  - to be very clear where function comes from
  - if same function name in different packages
- Examples:

```
blockclustr::blockclustr()
dplyr::left_join()
rvest::read_html()
```

Or access without loading

- Syntax is ‘`package::function()`
- Grabs function from the package
- Useful:
  - to be very clear where function comes from
  - if same function name in different packages
- Examples:

```
blockclustr::blockclustr()
dplyr::left_join()
rvest::read_html()
```

Or access without loading

- Syntax is ‘package::function()
- Grabs function from the package
- Useful:
  - to be very clear where function comes from
  - if same function name in different packages
- Examples:

```
blockclustr::blockclustr()
dplyr::left_join()
rvest::read_html()
```

Or access without loading

- Syntax is ‘`package::function()`
- Grabs function from the package
- Useful:
  - to be very clear where function comes from
  - if same function name in different packages
- Examples:

```
blockclustr::blockclustr()
dplyr::left_join()
rvest::read_html()
```

Seeing the nuts and bolts

- Lots of people make R packages
- There are pros and cons of this. Thoughts?
  - Some are better than others. . .
  - Sometimes they are wrong. Sometimes they break.
- You might need to dig into the code itself
- On GitHub, it's easy
  - Example: [https://github.com/erossiter/blockclustr]
- On CRAN, it's easy too, you just have to know where to look
  - Example: [https://cran.r-project.org/web/packages/stm/index.html]

Seeing the nuts and bolts

- Lots of people make R packages
- There are pros and cons of this. Thoughts?
  - Some are better than others. . .
  - Sometimes they are wrong. Sometimes they break.
- You might need to dig into the code itself
- On GitHub, it's easy
  - Example: [https://github.com/erossiter/blockclustr]
- On CRAN, it's easy too, you just have to know where to look
  - Example: [https://cran.r-project.org/web/packages/stm/index.html]

Seeing the nuts and bolts

- Lots of people make R packages
- There are pros and cons of this. Thoughts?
  - Some are better than others. . .
  - Sometimes they are wrong. Sometimes they break.
- You might need to dig into the code itself
- On GitHub, it's easy
  - Example: [https://github.com/erossiter/blockclustr]
- On CRAN, it's easy too, you just have to know where to look
  - Example: [https://cran.r-project.org/web/packages/stm/index.html]

Seeing the nuts and bolts

- Lots of people make R packages
- There are pros and cons of this. Thoughts?
  - Some are better than others. . .
  - Sometimes they are wrong. Sometimes they break.
- You might need to dig into the code itself
- On GitHub, it's easy
  - Example: [https://github.com/erossiter/blockclustr]
- On CRAN, it's easy too, you just have to know where to look
  - Example: [https://cran.r-project.org/web/packages/stm/index.html]

Seeing the nuts and bolts

- Lots of people make R packages
- There are pros and cons of this. Thoughts?
  - Some are better than others. . .
  - Sometimes they are wrong. Sometimes they break.
- You might need to dig into the code itself
- On GitHub, it's easy
  - Example: [https://github.com/erossiter/blockclustr]
- On CRAN, it's easy too, you just have to know where to look
  - Example: [https://cran.r-project.org/web/packages/stm/index.html]

Seeing the nuts and bolts

- Lots of people make R packages
- There are pros and cons of this. Thoughts?
  - Some are better than others. . .
  - Sometimes they are wrong. Sometimes they break.
- You might need to dig into the code itself
- On GitHub, it's easy
  - Example: [https://github.com/erossiter/blockclustr]
- On CRAN, it's easy too, you just have to know where to look
  - Example: [https://cran.r-project.org/web/packages/stm/index.html]

Seeing the nuts and bolts

- Lots of people make R packages
- There are pros and cons of this. Thoughts?
  - Some are better than others...
  - Sometimes they are wrong. Sometimes they break.
- You might need to dig into the code itself
- On GitHub, it's easy
  - Example: [https://github.com/erossiter/blockclustr]
- On CRAN, it's easy too, you just have to know where to look
  - Example: [https://cran.r-project.org/web/packages/stm/index.html]

Seeing the nuts and bolts

- Lots of people make R packages
- There are pros and cons of this. Thoughts?
  - Some are better than others. . .
  - Sometimes they are wrong. Sometimes they break.
- You might need to dig into the code itself
- On GitHub, it's easy
  - Example: [https://github.com/erossiter/blockclustr]
- On CRAN, it's easy too, you just have to know where to look
  - Example: [https://cran.r-project.org/web/packages/stm/index.html]

Seeing the nuts and bolts

- Lots of people make R packages
- There are pros and cons of this. Thoughts?
  - Some are better than others. . .
  - Sometimes they are wrong. Sometimes they break.
- You might need to dig into the code itself
- On GitHub, it's easy
  - Example: [https://github.com/erossiter/blockclustr]
- On CRAN, it's easy too, you just have to know where to look
  - Example: [https://cran.r-project.org/web/packages/stm/index.html]

Activity

quanteda: Text-as-data

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - “bag of words”
    - order doesn’t matter
  - stemming
  - remove punctuation, numbers, “stopwords,” most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - "bag of words"
    - order doesn't matter
  - stemming
  - remove punctuation, numbers, "stopwords," most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - "bag of words"
    - order doesn't matter
  - stemming
  - remove punctuation, numbers, "stopwords," most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - "bag of words"
    - order doesn't matter
  - stemming
  - remove punctuation, numbers, "stopwords," most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - "bag of words"
    - order doesn't matter
  - stemming
  - remove punctuation, numbers, "stopwords," most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - “bag of words”
    - order doesn’t matter
  - stemming
  - remove punctuation, numbers, “stopwords,” most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - "bag of words"
    - order doesn't matter
  - stemming
  - remove punctuation, numbers, "stopwords," most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - "bag of words"
    - order doesn't matter
  - stemming
  - remove punctuation, numbers, "stopwords," most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - "bag of words"
    - order doesn't matter
  - stemming
  - remove punctuation, numbers, "stopwords," most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - “bag of words”
    - order doesn’t matter
  - stemming
  - remove punctuation, numbers, “stopwords,” most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - "bag of words"
    - order doesn't matter
  - stemming
  - remove punctuation, numbers, "stopwords," most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - “bag of words”
    - order doesn’t matter
  - stemming
  - remove punctuation, numbers, “stopwords,” most frequent or infrequent words

Representing text as numbers

- We usually have a **corpus** of **documents**
  - Each document is made up of text!
- We need to represent the text in some quantitative way
  - **Document-term matrix** (DTM)
  - (Board)
- Usual simplifying assumptions:
  - n-gram
    - usually 1-gram
  - "bag of words"
    - order doesn't matter
  - stemming
  - remove punctuation, numbers, "stopwords," most frequent or infrequent words

When to do this vs. stringr?

- Representing text as a datset is useful for subsequent, systematic analysis
  - what proportion of documents talk about X?
  - what are the main topics across the Corpus?
- What we did with stringr was aimed at data cleaning
  - Lots of ways emails, names, address are in my dataset, I need to make variable systematic
  - etc.
- Do not reinvent the wheel. Get your document into a DTM as soon as possible if you're going to do analyses with text.

When to do this vs. stringr?

- Representing text as a datset is useful for subsequent, systematic analysis
  - what proportion of documents talk about X?
  - what are the main topics across the Corpus?
- What we did with stringr was aimed at data cleaning
  - Lots of ways emails, names, address are in my dataset, I need to make variable systematic
  - etc.
- Do not reinvent the wheel. Get your document into a DTM as soon as possible if you're going to do analyses with text.

When to do this vs. stringr?

- Representing text as a datset is useful for subsequent, systematic analysis
  - what proportion of documents talk about X?
  - what are the main topics across the Corpus?
- What we did with stringr was aimed at data cleaning
  - Lots of ways emails, names, address are in my dataset, I need to make variable systematic
  - etc.
- Do not reinvent the wheel. Get your document into a DTM as soon as possible if you're going to do analyses with text.

When to do this vs. stringr?

- Representing text as a datset is useful for subsequent, systematic analysis
  - what proportion of documents talk about X?
  - what are the main topics across the Corpus?
- What we did with stringr was aimed at data cleaning
  - Lots of ways emails, names, address are in my dataset, I need to make variable systematic
  - etc.
- Do not reinvent the wheel. Get your document into a DTM as soon as possible if you're going to do analyses with text.

When to do this vs. stringr?

- Representing text as a datset is useful for subsequent, systematic analysis
  - what proportion of documents talk about X?
  - what are the main topics across the Corpus?
- What we did with stringr was aimed at data cleaning
  - Lots of ways emails, names, address are in my dataset, I need to make variable systematic
  - etc.
- Do not reinvent the wheel. Get your document into a DTM as soon as possible if you're going to do analyses with text.

When to do this vs. stringr?

- Representing text as a datset is useful for subsequent, systematic analysis
  - what proportion of documents talk about X?
  - what are the main topics across the Corpus?
- What we did with stringr was aimed at data cleaning
  - Lots of ways emails, names, address are in my dataset, I need to make variable systematic
  - etc.
- Do not reinvent the wheel. Get your document into a DTM as soon as possible if you're going to do analyses with text.

When to do this vs. stringr?

- Representing text as a datset is useful for subsequent, systematic analysis
  - what proportion of documents talk about X?
  - what are the main topics across the Corpus?
- What we did with stringr was aimed at data cleaning
  - Lots of ways emails, names, address are in my dataset, I need to make variable systematic
  - etc.
- Do not reinvent the wheel. Get your document into a DTM as soon as possible if you're going to do analyses with text.

Example script

Rcpp: Integrating R and C++

Making R faster

- Rcpp is a way to integrate C++ code into R functions
- Why would we want to do this?
  - C++ is much, much faster
- Deep down R is actually C
- R is a high level language designed to make **coding** faster and easier, but... as a consequence it can often execute slowly
- Hadley Wickham yet again has an amazing book on this

Making R faster

- Rcpp is a way to integrate C++ code into R functions
- Why would we want to do this?
  - C++ is much, much faster
- Deep down R is actually C
- R is a high level language designed to make **coding** faster and easier, but... as a consequence it can often execute slowly
- Hadley Wickham yet again has an amazing book on this

Making R faster

- Rcpp is a way to integrate C++ code into R functions
- Why would we want to do this?
  - C++ is much, much faster
- Deep down R is actually C
- R is a high level language designed to make **coding** faster and easier, but. . . as a consequence it can often execute slowly
- Hadley Wickham yet again has an amazing book on this

Making R faster

- Rcpp is a way to integrate C++ code into R functions
- Why would we want to do this?
  - » C++ is much, much faster
- Deep down R is actually C
- R is a high level language designed to make **coding** faster and easier, but… as a consequence it can often execute slowly
- Hadley Wickham yet again has an amazing book on this

Making R faster

- Rcpp is a way to integrate C++ code into R functions
- Why would we want to do this?
  - C++ is much, much faster
- Deep down R is actually C
- R is a high level language designed to make **coding** faster and easier, but... as a consequence it can often execute slowly
- Hadley Wickham yet again has an amazing book on this

Making R faster

- Rcpp is a way to integrate C++ code into R functions
- Why would we want to do this?
  - C++ is much, much faster
- Deep down R is actually C
- R is a high level language designed to make **coding** faster and easier, but... as a consequence it can often execute slowly
- Hadley Wickham yet again has an amazing book on this

At a high level

```
library(Rcpp)
cppFunction(
'int addC(int x, int y, int z) {
  int sum = x + y + z;
            return sum;
            }')
addC(1,2,3)
```

```
## [1] 6
```

- You must declare the type for
  - inputs
  - outputs
  - variables

Compared to R…

In R things are quite different! What's different?

```r
addR <- function(x, y, z){
  return(x + y +z)
}
```

Exposure topic because…

- You need a C++ compiler to use `Rcpp` package
- You might have it, so feel free to try following along, but I won't debug in class