

[illegible]

1/5

- RStudio
- others (GitHub desktop etc.)

Basic Git workflow with RStudio

- create **repository** on Github
 - copy repository to local machine
 - git clone
 - RStudio: File > New Project > Version Control > Git > fill in name from "Clone" button on GH
-
- repeat:
 - **pull** (fetch and integrate changes from GH) [git pull]
 - RStudio: Git panel > click blue down-arrow
 - do stuff (create, edit files, etc.)
 - **stage** [git add]
 - RStudio: Git panel > click "Staged" button
 - **commit** [git commit]
 - RStudio: Git panel > click "Commit" icon > enter commit message > click "Commit" button (*ignore "amend previous commit" button!*)
 - **push** [git push]
 - RStudio: Git panel > click green up-arrow

tidyverse

- set of R packages: <https://www.tidyverse.org/>
- advantages
 - expressiveness
 - speed
 - new hotness
- disadvantages
 - minor incompatibilities with base R
 - rapid evolution
 - non-standard evaluation

tidyverse: big ideas

- new verbs
- piping
- tibbles

tidyverse: new verbs

- `filter(x,condition)`: choose rows equivalent to `subset(x,condition)` or `x[condition,]` (with non-standard evaluation)
- `select(x,condition)`: choose columns
 - equivalent to `subset(x,select=condition)` or `x[,condition]`
 - helper functions such as `starts_with()`, `matches()`
- `mutate(x,var=...)`: change or add variables (equivalent to `x$var = ...` or `transform(x,var=...)`)

tidyverse: split-apply-combine

- `group_by()`: adds grouping information
- `summarise()`: collapses variables to a single value
- e.g.

```
x <- group_by(x,course)
summarise(x,mean_score=mean(score),sd_score=sd(score))
```

- equivalent to

```
d_split <- split(d,d$var)      ## split
d_proc <- lapply(d_split, ...) ## apply
d_res <- do.call(rbind,d_proc) ## combine
```

tidyverse: piping

- new `%>%` operator (orig. from `magrittr` package)
- directs result of previous operation to next function, as first argument
- e.g.

```
(d_input
  %>% select(row1,row2)
  %>% filter(cond1,cond2)
  %>% mutate(...)) -> d_output
```

tidyverse: tibbles

- extension of data frames
- differences
 - printing
 - only prints first few rows/columns
 - labels columns by type
 - no rownames

- never drops dimensions (`tib[, "column1"]` is still a tibble)

tidyverse: reshaping (tidyr package)

- `gather(data, key, value, <include/exclude>)`
 - wide to long
 - `reshape2::melt()`
 - `pivot_longer()` in tidyr v 1.0 (see [here](#))
- `spread(data, key, value)`
 - long to wide
 - `reshape2::cast()`
 - `pivot_wider()` in tidyr v 1.0

types of data visualization

exploratory

- find patterns in data, explore hypotheses
- emphasize **robust** approaches
- minimize (parametric) assumptions
- [John Tukey](#), [William Cleveland](#) 1993.

diagnostic

- evaluate assumptions of a model
 - unbiasedness/goodness of fit
 - homoscedasticity
 - normality
- easily spot deviations
- identify outliers and influential points

inferential

- coefficient plots (e.g. `dotwhisker` package)
- replacement for tables (Gelman et al. 2002)
- also: tests of inference (Wickham et al. 2010)
- [Andrew Gelman](#)

expository: data-viz

- tell an accurate story
- high information density
- Cleveland, [Edward Tufte](#)

presentation: info-viz

- grab attention/engage/sell/entertain
- "puzzle" graphics

dashboards

- present a quick overview of a data set
- user control
- business-oriented

dynamic

- time dimension
- engage
- allow viewer to drill down
- [Dianne Cook](#)

References

Cleveland, W. 1993. *Visualizing Data*. Summit, NJ: Hobart Press.

Gelman, A et al. 2002. *The American Statistician* 56 (2): 121–130.
<http://www.tandfonline.com/doi/abs/10.1198/000313002317572790>.

Wickham, H et al. 2010. *IEEE Transactions on Visualization and Computer Graphics* 16 (6) (November): 973–979. doi:[10.1109/TVCG.2010.161](https://doi.org/10.1109/TVCG.2010.161).

[Course home page](#)