Coursera: Reproducible Research

Contents

[Reproducible Research 3](#_Toc424581865)

[Developing the “Score” for Reproducible Research 3](#_Toc424581866)

[Concepts and Ideas 3](#_Toc424581867)

[Replication 3](#_Toc424581868)

[What’s Wrong with Replication? 3](#_Toc424581869)

[Why Do We Need Reproducible Research? 3](#_Toc424581870)

[Example: Reproducible Air Pollution and Health Research 3](#_Toc424581871)

[Research Pipeline 4](#_Toc424581872)

[Recent Developments in Reproducible Research 4](#_Toc424581873)

[What do We Need? 4](#_Toc424581874)

[Who are the players? 5](#_Toc424581875)

[Challenges 5](#_Toc424581876)

[In Reality… 5](#_Toc424581877)

[Sweave Limitations 5](#_Toc424581878)

[Knitr 6](#_Toc424581879)

[Summary 6](#_Toc424581880)

[Structures of Data Analysis 6](#_Toc424581881)

[Steps in a data analysis 6](#_Toc424581882)

[The key challenge in data analysis 6](#_Toc424581883)

[Defining a question 6](#_Toc424581884)

[An example 7](#_Toc424581885)

[Define the ideal data set 7](#_Toc424581886)

[Determine what data you can access 7](#_Toc424581887)

[Obtain the data 7](#_Toc424581888)

[Clean the data 8](#_Toc424581889)

[Subsampling our data set 8](#_Toc424581890)

[Exploratory data analysis 8](#_Toc424581891)

[Statistical prediction/modeling 8](#_Toc424581892)

[Interpret results 8](#_Toc424581893)

[Our example 8](#_Toc424581894)

[Challenge results 8](#_Toc424581895)

[Synthesize/write-up results 9](#_Toc424581896)

[Our example 9](#_Toc424581897)

[Create reproducible code 9](#_Toc424581898)

[Organizing a Data Analysis 9](#_Toc424581899)

[Data analysis files 9](#_Toc424581900)

[Raw data 10](#_Toc424581901)

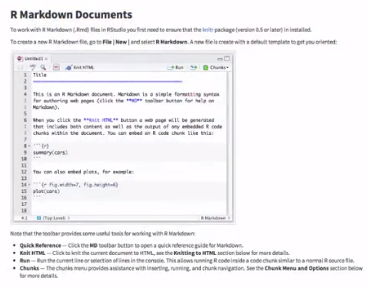
[Processed data 10](#_Toc424581902)

[Exploratory figures 10](#_Toc424581903)

[Final figures 10](#_Toc424581904)

[Raw scripts 10](#_Toc424581905)

[Final scripts 11](#_Toc424581906)

[R markdown files  11](#_Toc424581907)

[Readme files 11](#_Toc424581908)

[Text of the document 12](#_Toc424581909)

[Further resources 12](#_Toc424581910)

# Reproducible Research

Basic idea is when you’re doing data analysis with many different steps, when you communicate what you’ve done, that’s reproducible by someone else. Subsequent analysis are not reproducible by someone else.

# Developing the “Score” for Reproducible Research

There are a variety of ways to communicate data analysis, but we haven’t agreed on a way for everyone to do this. In this course we will focus on how to communicate a data analysis using code by writing documents that are dynamic.

# Concepts and Ideas

## Replication

* The ultimate standard for strengthening scientific evidence is replication of findings and conducting studies with independent
  + Investigators
  + Data
  + Analytical methods
  + Laboratories
  + Instruments
* Replication is particularly important in studies that can impact broad policy or regulatory decisions

## What’s Wrong with Replication?

* Some studies cannot be replicated
  + No time, opportunistic
  + No money
  + Unique
* Reproducible Research: Make analytic data and code available so that others may reproduce findings

How can we Bridge the Gap?

Replication & Nothing 🡪 Reproducible

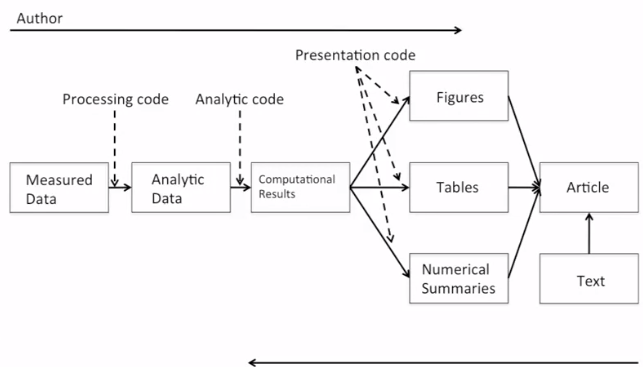
## Why Do We Need Reproducible Research?

* New technologies increasing data collection throughput; data are more complex and extremely high dimensional
* Existing databases can be merged into new “megadatabase”
* Computing power is greatly increased, allowing more sophisticated analyses
* For every field “X” there is a field “Computational X” (e.g. computation biology)

## Example: Reproducible Air Pollution and Health Research

* Estimating small (but important) health effects in the presence of much stronger signals (e.g. smoking is more harmful than pollution)
* Results inform substantial policy decisions, affect many stakeholders
  + EPA regulations can cost billions of dollars
* Complex statistical methods are needed and subjected to intense scrutinty
* See Internet-based Health and Air Pollution Surveillance System (iHAPSS)

# Research Pipeline



## Recent Developments in Reproducible Research

* Science Data Replication & Reproducibility
* The Duke Saga (60 Minutes)
* Institute of Medicine – Evolution of Translational Omics
  + Data/metadata used to develop tests should be made publicly available
  + The computer code and fully specified computation procedures used for development of the candidate omics-based test should be made sustainably available
  + “Ideally, the computer code that is released will **encompass all of the steps of computation analysis,** including all data preprocessing steps, that have been described in tis chapter. All aspects of the analysis need to be transparently reported.”

## What do We Need?

* Analytic data are available
* Analytic code are available
* Documentation of code and data
* Standard means of distribution

## Who are the players?

* Authors
  + Want to make their research reproducible?
  + Want tools for RR to make their lives easier (or at least not much harder)
* Readers
  + Want to reproduce (and perhaps expand upon)
  + Want tools to make their lives easer

## Challenges

* Authors must undertake considerable efforts to put data/results on the web (may not have resources like a web server)
* Readers must download data/results individually and piece together which data go with which code sections, etc.
* Readers may not have the same resources as authors
* Few tools to help authors/readers (although toolbox is growing!)

## In Reality…

* Authors
  + Just put stuff on the web
  + (Infamous) Journal supplementary materials
  + There are some central databases for various fields (e.g. biology, ICPSR)
* Readers
  + Just download the data and (try to) figure it out
  + Piece together the software and run it

Literature (Statistical) Programming

* An article is a stream of **text** and **code**
* Analysis code is divided into text and code “chunks”
* Each code chunk loads data and computes results
* Presentation code formats results (tables, figures, etc.)
* Article text explaining what is going on
* Literate programs can be **weaved** to produce human-readable documents and **tangled** to produce machine readable code
* Literature programming is a general concept that requires
  1. A documentation language (human readable)
  2. A programming language (machine readable)
* Sweave uses Latext and R as the documentation and programming languages
* Sweave was developed by Friedrich Leisch (member of the R Core) and is maintained by R Core
* Main web site: <http://www.statistik.lmu.de/~leisch/Sweave>

## Sweave Limitations

* Sweave has many limitations
* Focused primarily on Latex, a difficult to learn markup language used only by weirdos
* Lacks features like caching, multiple plots per chunk, mixing programming languages and many other technical items
* Not frequently updated or very actively developed

## Knitr

* Knitr is an alternative (more recent) package
* Brings together many features added on to Sweave to address limitations
* Knitr uses R as the programming language (although others are allowed) and variety of documentation languages
  + LaTeX, Markdown, HTML
* Knitr was developed by Yihui Xie (while a graduate student in statistics at Iowa State)
* See <http://yihui.name/knitr>

## Summary

* Reproducible research is important as a **minimum standard**, particularly for studies that are difficult to replicate
* Infrastructure is needed for **creating** and **distributing** reproducible documents, beyond what is currently available
* There is a growing number of tools for creating reproducible documents

# Structures of Data Analysis

## Steps in a data analysis

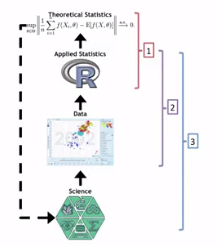
* Define the questions
* Define the ideal data set
* Determine what data you can access
* Obtain the data
* Clean the data
* Exploratory data analysis
* Statistical prediction/modeling
* Interpret results
* Challenge results
* Synthesize/write up results
* Create reproducible code

## The key challenge in data analysis

“Ask yourself, what problem have you solved, ever, that was worth solving, where you knew all the given information in advance? Where you didn’t have a surplus of information and have to filter it out, or you had insufficient information and have to go find some?” Dan Myer, Mathematics Educator

## Defining a question

1. Statistical methods development
2. Danger zone!!! – application of statistics to raw data without any science
3. Proper data analysis



## An example

**Start with a general question**

Can I automatically detect emails that are SPAM that are not?

**Make it concrete**

Can I use quantitative characteristics of the emails to classify them as SPAM/HAM?

## Define the ideal data set

* The data set may depend on your goal
  + Descriptive – a whole population
  + Exploratory – a random sample with many variables measured
  + Inferential – the right population, randomly sampled
  + Predictive – training and test data set from the same population
  + Causal – data from a randomized study
  + Mechanistic – data about all components of the system

## Determine what data you can access

* Sometimes you can find data free on the web
* Other times you may need to buy the data
* Be sure to respect the terms of use
* If the data don’t exist, you may need to generate it yourself
* A possible solution: <http://archive.ics.uci.edu/ml/datasets/spambase>

## Obtain the data

* Try to obtain the raw data
* Be sure to reference the source
* Polite emails go a long way
* If you will load the data from an internet source, record the url and time accessed

Our data set <http://search.r-project.org/library/kernlab/html/spam.html>

## Clean the data

* Raw data often needs to be processed
* If it is pre-processed, make sure you understand how
* Understand the source of the data (census, sample, convenience sample, etc.)
* May need reformatting, subsampling – record these steps
* **Determine if the data are good enough** - if not, quit or change data
* Do not push on with bad databecause this can lead to inappropriate influences or conclusions

## Subsampling our data set

We need to generate a test and training set (prediction)

## Exploratory data analysis

* Look at summaries of the data
* Check for missing data
* Create exploratory plots
* Perform exploratory analyses (e.g. clustering)

## Statistical prediction/modeling

* Should be informed by the results of your exploratory analysis
* Exact methods depend on the question of interest
* Transformations/processing should be accounted for when necessary
* Measures of uncertainty should be reported

## Interpret results

* Use the appropriate language
  + Describes
  + Correlated with/associated with
  + Leads to/causes
  + Predicts
* Give an explanation
* Interpret coefficients
* Interpret measures of uncertainty

### Our example

* The fraction of characters that are dollar signs can be used to predict if an email is Spam
* Anything with more than 6.6.% dollar signs is classified as Spam
* More dollar signs always means more Spam under our prediction
* Our test set error rate was 22.4%

## Challenge results

* Challenge all steps
  + Questions
  + Data source
  + Processing
  + Analysis
  + Conclusions
* Challenge measures of uncertainty
* Challenge choices of terms to include in models
* Think of potential alternatives analyses

## Synthesize/write-up results

* Lead with the question
* Summarize the analyses into the story
* Don’t include every analysis, include it
  + If it is needed for the story
  + If it is needed to address a challenge
* Order analyses according to the story, rather than chronologically
* Include “pretty” figures that contribute to the story

### Our example

* Lead with the questions
  + Can I use quantitative characteristics of the emails to classify them as SPAM/HAM?
* Describe the approach
  + Collected data from UCI -> created training/test sets
  + Explored relationships
  + Choose logistic model on training set by cross validation
  + Applied to text, 78% test set accuracy
* Interpret results
  + Number of dollar signs seems reasonable, e.g. “Make money with Viagra $ $ $ $!”
* Challenge results
  + 78% isn’t that great
  + I could use more variables
  + Maybe not logistic regression

## Create reproducible code

Use knitr, for example

# Organizing a Data Analysis

## Data analysis files

* Data
  + Raw data
  + Processed data
* Figures
  + Exploratory figures
  + Final figures
* R code
  + Raw / unused scripts
  + Final scripts
  + R Markdown files
* Text
  + README files
  + Text of analysis / report

## Raw data

* Should be stored in your analysis folder
* If accessed from the web, include url, description and data accessed in README
* If using git you can add raw data to repository if not too big, can provide information in log file

## Processed data

* Processed data should be named so it is easy to see which script generated the data
* The processing script – processed data mapping should occur in the README
* Processed data should be [tidy](http://vita.had.co.nz/papers/tidy-data.pdf)

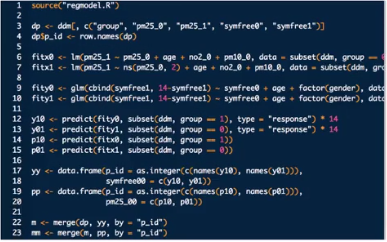
## Exploratory figures

* Figures made during the course of your analysis, not necessarily part of your final report
* They do not need to be “pretty”

## Final figures

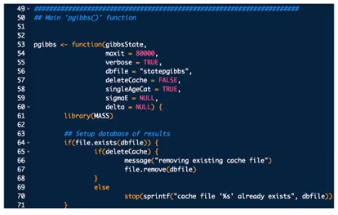
* Usually a small subset of the original figures
* Axes/colors set to make the figure clear
* Possibly multiple panels

## Raw scripts



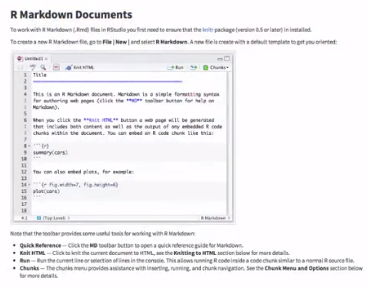
* May be less commented (but comments help you!)
* May be multiple versions
* May include analyses that are later discarded

## Final scripts



* Clearly commented
  + Small comments liberally – what, when, why, how
  + Bigger commented blocks for whole sections
* Include processing details
* Only analyses that appear in the final write-up

## R markdown files



* R markdown files can be used to generate rep
* Text and R code are integrated
* Very easy to create in Rstudio

## Readme files

* Not necessary if you use R markdown
* Should contain step-by-step instructions for analysis
* Here is an example <https://github.com/jtleek/swfdr/blob/master/README.md>

## Text of the document

* It should include a title, introduction (motivation), methods (statistics you used), results (including measures of uncertainty), and conclusions (including potential problems)
* It should tell a story
* *It should not include every analysis you performed*
* References should be included for statistical methods

## Further resources

* Information about a non-reproducible study that led to cancer patients being mistreated: [The Duke Saga Starter Set](http://simplystatistics.org/2012/02/27/the-duke-saga-starter-set/)
* [Reproducible research and Biostatistics](http://biostatistics.oxfordjournals.org/content/10/3/405.full)
* [Managing a statistical analysis project guidelines and best practices](http://www.r-statistics.com/2010/09/managing-a-statistical-analysis-project-guidelines-and-best-practices/)
* [Project template](http://projecttemplate.net/) - a pre-organized set of files for data analysis

# Coding Standards in R

1. Always use text files / text editor
2. Ident your code – to show how control of program goes
   1. Indenting improves readability
   2. Fixing line length (80 columns) prevents lots of nesting and very long functions
   3. Suggested: Indents of 4 spaces at minimum; 8 spaces ideal
3. Limit the width of your code (80 columns?)
   1. Insert carriage return AFTER the operator (chain, plus, etc.)
4. Limit the length of individual functions – one basic activity

# Markdown

Really useful language for formatting document. A simplified markup language. Very easy to integrate with R and other programming languages.

## What is Markdown?

“Markdown is a text-to-HTML conversion tool for web writers”

Markdown Syntax

|  |  |
| --- | --- |
| Command | Description |
| \*italic\* | sets up plot without the data |
| \*\*bold\*\* | **bold** |
| ## heading 1  ### heading 2 | Heading 1Heading 2 |
| * unordered list item * unordered list item |  |
| * 1. First item in list   2. Second item in list   3. Third item in list |  |
| [John Hopkins Bloomberg School of Public Health](http://www.jhsph.edu/) | [John Hopkins Bloomberg School of Public Health](http://www.jhsph.edu/) |
| I spend so much time reading [R Bloggers] [1] and [Simply Statistics] [2] !  [1]: <http://www.r-bloggers.com/> "R Bloggers"  [2]: <http://simplystatistics.org/> "Simply Statistics" | I spend so much time reading [R Bloggers](http://www.r-bloggers.com/) and [Simply Statistics](http://simplystatistics.org)! |
| First line requires two spaces for new line  Second line | First line requires two spaces for new line  Second line |

## Resources

* The Offical Markdown Documentation <http://daringfireball.net/projects/markdown/basics>
* Github's Markdown Guide - <https://help.github.com/articles/github-flavored-markdown>

# R Markdown

You want to have R code integrated with a simple language to

## What is Markdown?

* Created by John Gruber and Aaron Swartz
* A simplified version of “markup” languages
* Allows one to focus on writing as opposed to formatting
* Simple/minimal intuitive formatting elements
* Easily converted to valid HTML (and other formats) using existing tools
* Complete information available <http://daringfireball.net/projects/markdown/>
* Some background information at <http://daringfireball.net/2004/03/dive_into_markdown>

## What is R Markdown?

* R markdown is the integration of R code with markdown
* Allows one to create documents containing “live” R code
* R code is evaluated as part o the processing of the markdown
* Results from R code are inserted into markdown document
* A core tool in **literate statistical programming**
* R markdown can be converted to standard markdown using the knitr package in R
* Markdown can be converted to HTML using the markdown package in R
* Any basic text editor can be used to create a markdown document; no special editing tools needed
* You do not need to edit the markdown or HTML, only the R markdown
* The R markdown 🡪 markdown 🡪 HTML work flow can be easily managed using R Studio (but not required
* These slides were written in R markdown and converted to slides using the [slidify](http://slidify.org) package

# Literate Statistical Programming with knitr

## Problems, Problems

* Authors must undertake considerable effort to put data/results on the web
* Readers must download data/results individually and piece together which data go with which code sections, etc
* Authors/readers must manually interact with websites
* There is no single document to integrate data analysis with textual representations; i.e. data, code, and text are not linked

## Literate Statistical Programming

* Original idea comes from Don Knuth
* An article is a stream of text and code
* Analysis code is divided into text and code “chunks”
* Presentation code formats results (tables, figures, etc.)
* Article text explains what is going on
* Literate programs are weaved to produce human-readable documents or tangles to produce manchine-readable document
* Literate programming is a general concept. We need
  + A documentation language
  + A programming language
* The original Sweave system developed by Friedrich Leisch used LaTeX and R
* knitr supports a variety of documentation languages

## How Do I Make My Work Reproducible?

* Decide to do it (ideally from the start)
* Keep track of things, perhaps with a version control system to track snapshots/changes
* Use software whose operation can be coded (gui not ideal)
* Don’t save output (such as temporary data transformations – keep original data and process that got you there)
* Save data in non-proprietary formats

## Literate Programming: Pros

* Text and code all in one place, logical order
* Data, results automatically updated to reflect external changes
* Code is live—automatic “regression test” when building a document (running of live code will keep you from introducing new errors into analysis)

## Literate Programming: Cons

* Text and code all in one place, can make documents difficult to read, especially if there is a lot of code
* Can substantially slow down processing of documents (although there are tools to help)

## What is knitr?

* An R package written by Yihui Xie (while he was a grad student at Iowa State)
  + Available on CRAN
  + Supports RMarkdown

## Requirements

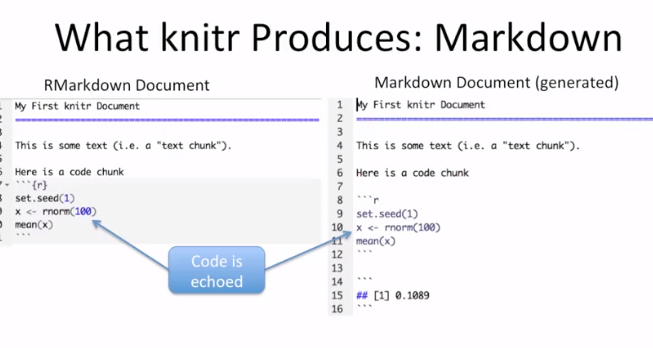
* A recent version of R
* A text editor (the one that comes with Rstudio is okay)
* Some support packages also available on CRAN
* Some knowledge of Markdown, LaTeX, HTML

## What is knitr Good For?

* Manuals
* Short/medium-length technical documents
* Tutorials
* Reports (esp. if generated periodically)
* Data preprocessing documents/summaries

## What is knitr NOT Good for?

* Very long research articles
* Complex time-consuming computations
* Documents that require precise formatting



* Knitr will fill a new document with filter text; delete it
* Code chunks begin with ```{r} and end with ```
* All R code goes in between these markers
* Code chunks can have names, which is useful when we start making graphics  
  ```{r firstchunk}  
  ### R Code Goes Here  
  ```
* By default will show code (echoed), if you do not want use, {r, echo=FALSE}

## Processing of knitr Documetns (what happens under the hood)

* You write the RMarkdown document (.Rmd)
* Knitr produces a Markdown document (.md)
* Knitr converts the Markdown document into HTML (by default)
* .Rmd -> .md >- .html
* Do not edit .md or .html

Inline Text Computations

## Setting Global Options

* Sometimes we want to set options for **every** code chunk that are different from the defaults
* For example, we may want to suppress all code echoing and results output
* We have to write some code to set these global options
* Introduce code chunk with:  
   opts\_chunk$set(echo=FALSE,results=”hide”)

## Some Common Options

* Output
  + Results: “asis”, “hide”
  + Echo: TRUE, FALSE
* Figures
  + Fig.height: numeric
  + Fig.width: numeric

## Caching Computations

* What if one chunk takes a long time to run?
* All chunks have to be re-computed every time you re-knit the file
* The cache=TRUE option can be set on a chunk-by-chunk basis to store results of computatiosn
* After the first run, results are loaded from cache

## Caching Caveats

* If the data or code (or anything external) changes you need to re-run the cached code chunks
* Dependencies are not checked explicitly
* Chunks with significant *side effects* may not be cacheable

## Summary

* Literate statistical programming can be a useful way to put text, code, data, output all in one document
* Knitr is a powerful tool for integrating code and text in a simple document format

# Communicating Results

## Tl;dr

* People are busy expecially managers and leaders
* Results of data analyses are sometimes presented in oral form, but often the first cut is presented via email
* It is often useful to breakdown the results of an a

## Hierarchy of Information: Research Paper

* Title / Author List
* Abstract
* Body / Results
* Supplementary Materials/ the gory details
* Code / Data / really gory details

## Hierarchy of Information: Email Presentation

* Subject line / Sender info
  + At a minimum; include one
  + Can you summarize findings in one sentence?
* Email body
  + A brief description of the problem / context; recall what was proposed and executed; summarize findings; results 1-2 paragraphs
  + If action needs to be taken as a result of this presentation, suggest some options and make them as concrete as possible
  + If questions need to be addressed, try to make them yes / no
* Attachements(s)
  + R Markdown file
  + Knitr report
  + Stay concise; don’t spit out pages of code (because you used knitr we knot it’s available)
* Links to Supplementary Materials
  + Code / Software / Data
  + GitHub repository / Project web site

## RPubs

* Little service that RStudio provides
* [www.Rpubs.com](http://www.Rpubs.com)
* Once you create an account you can publish markdown/knitr documents

# Reproducible Research Checklist

## DO: Start with Good Science

* Garbage in, garbage out
* Coherent, focused question simplifies many problems
* Working with good collaborators reinforces good practices
* Something that’s interesting to you will (hopefully) motivate good habits

## DON’T: Do Things By Hand

* Editing spreadsheets of data to “clean it up”
  + Removing outliers
  + QA / QC
  + Validating
* Editing tables or figures (e.g. rounding, formatting).
* Downloading data from a web site (clicking links in a web browser)
* Moving data around your computer; splitting / reformatting data files
* “We’re just going to do this once….”

Things done by hand need to be precisely documented (this is harder than it sounds)

## DON’T Point And Click

* Many data processing / statistical analysis packages have graphically user interfaces (GUIs)
* GUIs are convenient / intuitive but the actions you take with a GUI can be difficult for others to reproduce
* Some GUIs produce a log file or script which includes equivalent commands; these can be saved for later examination
* In general, be careful with data analysis software that is highly *interactive*; ease of use can sometimes lead to non-reproducible analyses
* Other interactive software, such as text editors, are usually fine

## DO: Teach a Computer

* If something needs to be done as part of your analysis / investigation, try to teach your computer to do it (even if you only need to do it once)
* In order to give your computer instructions, you need to write down exactly what you mean to do and how it should be done
* Teaching a computer almost guarantees reproducibility

**For example, by hand, you can**

1. Go to the UCI Machine Learning Repository at <http://archive.ics.uci.edu/ml>
2. Download the Bike Sharing Dataset by Clicking on the link to the Data Folder, then clinking on the link to the zip file of dataset, and choosing “Save Linked File As…” and then saving it to a folder on your computer

**Or you can teach your computer to do the same thing using R:**

download.file(“

Notice here that

* The full URL to the dataset file is specified (no clicking through a series of links)
* The name of the file saved to your local computer is specified
* The directory in which the file was saved is specified (“ProjectData”)
* Code can always be executed in R (as long as link is available)
* Code can always be executed in R (as long as link is available)

## DO: Use Some Version Control

* Slow things down
* Add changes in small chunks (don’t just do one massive commit)
* Track / tag snapshots; revert to old versions
* Software like GitHub / BitBucket / SourceForge make it easy to publish results

## DO: Keep Track of Your Software Environment

* If you work on a complex project involving many tools/datasets, the software and computing environment can be critical for reproducing your analysis
* **Computer architecture**: CPU (Intel, AMD, ARM) 32/64 bit; GPUs
* **Operating system**: Windows, Mac OS, Linux / Unix
* **Software toolchain**: Compilers, interpreters, command shell, programming languages (C, Perl, Python, etc.), database backends, data analysis software
* **Supporting software / infrastructures**: Libraries, R packages, dependencies
* **External dependencies**: Web sites, data repositories, remote databases, software repositories
* **Version numbers**: Ideally, for everything (if available)

## DON’T: Save Output

* Avoid saving data analysis output (tables, figures, summaries, processed data, etc.) except perhaps temporarily for efficiency purposes
* If a stray output file cannot be easily connected with the means by which it was created, then it is not reproducible
* Save the data + code that generated the output, rather than the output itself
* Intermediate files are okay as long as there is clear documentation of how they were created

## DO: Set Your Seed

* Random number generators generate pseudo-random numbers based on an initial seed (usually a number or set of numbers)
  + In R you can use the set.seed() function to set the seed and to specify the random number generator to use
* Setting the seed allows for the stream of random numbers to be exactly reproducible
* Whenever you generate random numbers for a non-trivial purpose, **always set the seed**

## DO: Think About the Entire Pipeline

* Data analysis is a lengthy process; it is not just tables / figures / reports
* Raw data -> processed data -> analysis -> report
* How you got the end is just as important as the end itself
* The more of the data analysis pipeline you can make reproducible, the better for everyone

## Summary: Checklist

* Are we doing good science?
* Was any part of this analysis done by hand?
  + If so, are those parts *precisely*
  + Does the documentation match reality?
* Have we taught a computer to do as much as possible (i.e. coded)?
* Are we using a version control system?
* Have we documented our software environment?
* Have we saved any output that we cannot reconstruct from original data + code?
* How far back in the analysis pipeline can we go before our results are no longer (automatically) reproducible?

# Reproducible Research with Evidence-based Data Analysis

**Replication**

* Focuses on the validity of the scientific claim
* “Is this claim true?”
* The ultimate standard for strengthening scientific evidence
* New investigators, data, analytical methods, laboratories, instruments, etc.
* Particularly important in studies that can impact broad policy or regulatory decisions

**Reproducibility**

* Focused on the validity of the data analysis
* “Can we trust this analysis?”
* Arguably a minimum standard for any scientific study
* New investigators, same data, same methods
* Important when replication is impossible

## Background and Underlying Trends

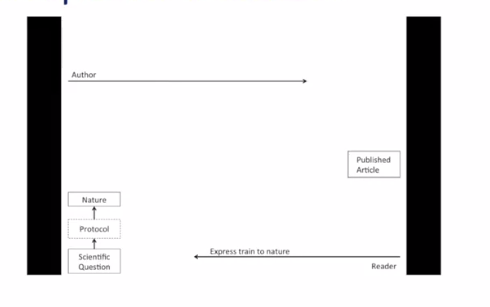
* Some studies cannot be replicated: No time, no money, unique/opportunistic
* Technology is increasing data collection throughput; data are more complex and high-dimensional
* Existing databases can be merged to become bigger databases (but data are used off-label)
* Computing power allows more sophisticated analyses, even on “small” data
* For every field “X” there is a “Computational X”

## The Result?

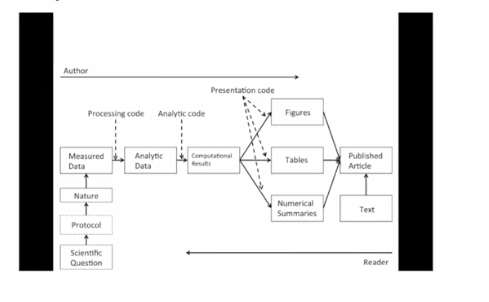
* Even basic analyses are difficult to describe
* Heavy computational requirements are trust upon people without adequate training in statistics and computing
* Errors are more easily introduced into analysis pipelines
* Knowledge transfer is inhibited
* Results are difficult to replicate or reproduce
* Sense that: Complicated results cannot be trusted

## What is Reproducible Research?

Traditionally:



Now:



## What Problem Does Reproducibility Solve?

**What we get**

* Transparency
* Data availability
* Software / methods availability
* Improved transfer of knowledge

**What we do NOT get**

* Validity/Correctness of the analysis

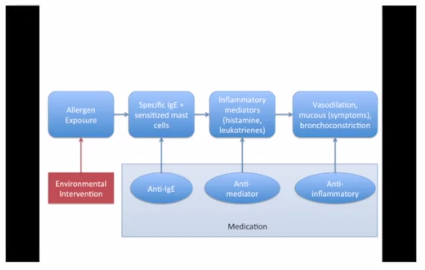
An analysis can be reproducible and still be wrong. We want to know “can we trust analysis?” Does requiring reproducibility deter bad analysis?

## Problems with Reproducibility

The premise of reproducible research is that with data/code available, people can check each other and the whole system is self-correcting

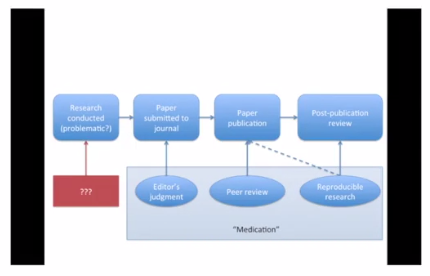
* Addressing the most “downstream” aspect of the research process – post-publication
* Assumes everyone plays by the same rules and wants to achieve the same goals (i.e. scientific discovery)

## An Analogy from Asthma



The upstream solution – environmental intervention – the cheapest and most effective solution. So what is the analogy to the issue of problematic research.

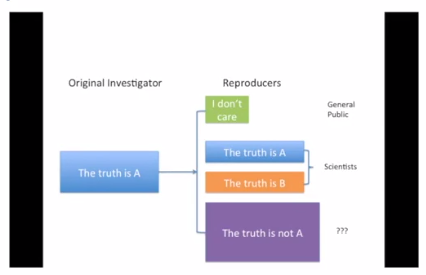
## Scientific Dissemination Process

****

At Biostatistics they are doing reproducibility in the Paper publication process, but this is just one model.

## Who Reproduces Research?

* For reproducibility to be effective as a means to check validity, someone needs to do something
  + Re-run the analysis; check results match
  + Check the code for bugs/errors
  + Try alternate approaches; check sensitivity
* The need for someone to do something is inherited from traditional notion of replication
* Who is “someone” and what are their goals?



Size of boxes is relative to likelihood of reproducing it.

## The Story So Far

* Reproducibility brings transparency (wrt code+data) and increased transfer of knowledge
* A lot of discussion about how to get people to share data
* Key question of “can we trust this analysis?” is not addressed by reproducibility
* Reproducibility addresses potential problems long after they’ve occurred (“downstream”)
* Secondary analyses are inevitably coloured by the interests/motivations of others

## Evidence-based Data Analysis

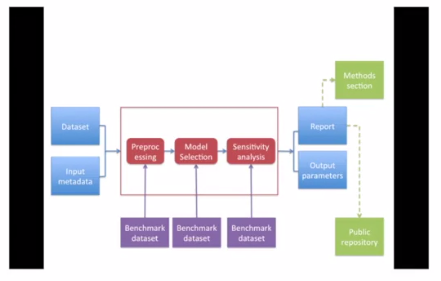
* Most data analyses involve stringing together many different tools and methods
* Some methods may be standard for a given field, but others are often applied ad hoc
* We should apply thoroughly studied (via **statistical research**), **mutually agreed upon methods** to analyze data whenever possible
* There should be evidence to justify the application of a given method

Question: How did binwidth in default histogram be chosen? Sturges HA (1926), Scott DW (1979). No one argues for the most part.

Argument; take this principle and apply it everywhere

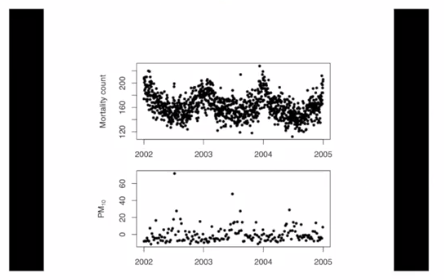
* Create analytic pipelines for evidence-based components – standardize it
* A Deterministic Statistical Machine <http://goo.gl/Qvlhuv>
* Once an evidence-based analytic pipeline is established, we shouldn’t mess with it
* Analysis with a “transparent box”
* Reduce the “researcher degrees of freedom”
* Analogous to a pre-specified clinical trial protocol

## Deterministic Statistical Machine

****

## Case Study: estimate Acute Effects Of Ambient Air Pollution Exposure

* Acute/short-term effects typically estimates via panel studies or time series studies
* Work originated in late 1970s early 1980s
* Key question: “Are short-term changes in population associated with short-term changes in a population health outcome
* Studies usually conducted at community level
* Long history of statistical research investigating proper methods of analysis



Objective: is top thing correlated with bottom thing

* Can we encode everything that we have found in statistical/epidemiological research into a single package?
* Time series studies do not have a huge range of variation; typically involves similar types of data and similar questions
* We can create a deterministic statistical machine for this area?

## DSM Modules for Time Series Studies of Air Pollution and Health

* Check for outliers, high leverage, overdispersion
* Fill in missing data? NO! (can try to impute it, but add a lot of noise for a little bit of savings on bias)
* Model selection: Estimate degrees of freedom to adjust for unmeasured confounders
  + Other aspects of model not as critical
* Multiple lag analysis
* Sensitivity analysis wrt
  + Unmeasured confounder adjustment
  + Influential Points

Some things, the way it is done do not matter: adjusting for temperature, weather doesn’t really matter.

## Where to Go From Here?

* One DSM is not enough, we need many!
* Different problems warrant different approaches and expertise
* A curated library of machines providing state-of-the-art analysis pipelines
* A CRAN/CPAN/CTAN/.. for data analysis
* Or a “Cochrane Collaboration” for data analysis

## A Model: Cochrane Collaboration

Meta-analysis on clinical trials to promote evidence based medicine.

## A Curated Library of Data Analysis

* Provide packages that encode data analysis pipelines for given problems, technologies, questions
* Curated by experts knowledgeable in the field
* Documentation/references given supporting each module in the pipeline
* Changes introduced after passing relevant benchmark/unit tests

## Summary

* Reproducible research is important, but does not necessarily solve the critical question of whether data analysis is trustworthy
* Reproducible research focused on the most “downstream” aspect of research dissemination
* Evidence-based data analysis would provide standardized, best practices for given scientific areas and questions
* Gives reviewers an important tool without dramatically increasing the burden on them
* More effort should be put into improving the quality of “upstream” aspects of scientific research