

# CONTAINERS

# OUTLINE

- The problem: dependencies
- Possible solutions
- Containers
  - What they are
  - Why use them
- Brief tutorial
- Additional advantages of containers
- Discussion

# **THE PROBLEM: DEPENDENCIES**

# **A FIRST ATTEMPT AT REPRODUCIBILITY (C. 2018)**

1. Create an R package for the method
2. Create an R package for data and helper routines
3. Create scripts to run the analyses
4. Put everything on github

## THEN TIME PASSES...

- When revising the paper, we updated our code, re-ran the analysis, and... got very different results for a method we had compared against
- After a great deal of debugging, we discovered a dependency of a dependency of a dependency had changed
- Moral: Saving your code is not enough. You need to save the entire computational environment

# EXAMPLE: lme4

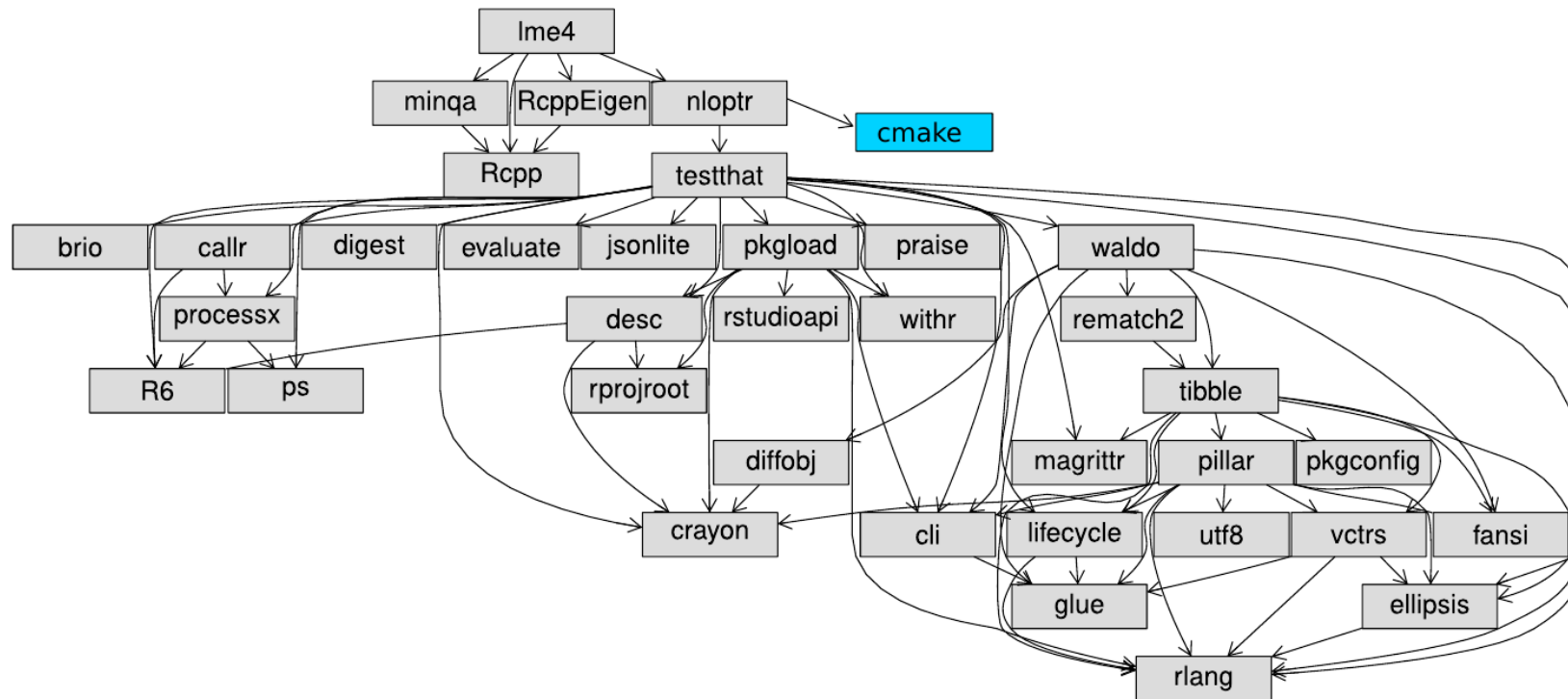


Figure 1: Dependency graph for the R package **lme4**. Grey boxes are R add-on packages. Arrows indicate dependency. The blue box indicates the system-level dependency of the package for Linux OS Ubuntu ver. 20.04.

# POSSIBLE SOLUTIONS

- Python: `venv`
- R: `renv`
- Containers (e.g., Podman, Docker)
- Others?

# ANOTHER EXAMPLE (C. 2020) – NOT EVEN A “DEPENDENCY”!

- Writing a paper with a student that analyzed social media data (Tweets)
- The student created a full analysis pipeline and shared on github
- Fairly simple pipeline, so waited to containerize until final revisions complete
- Time passes... paper accepted, time to containerize
- One figure in the paper: Randomly selected example tweets
- They changed! (And one was now very offensive)
- The method of random number generation for the `sample` command had changed. See: [link](#)

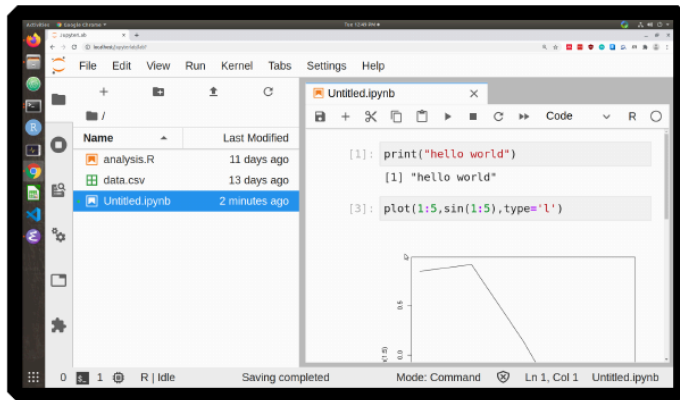


# CONTAINERS

# WHAT ARE CONTAINERS

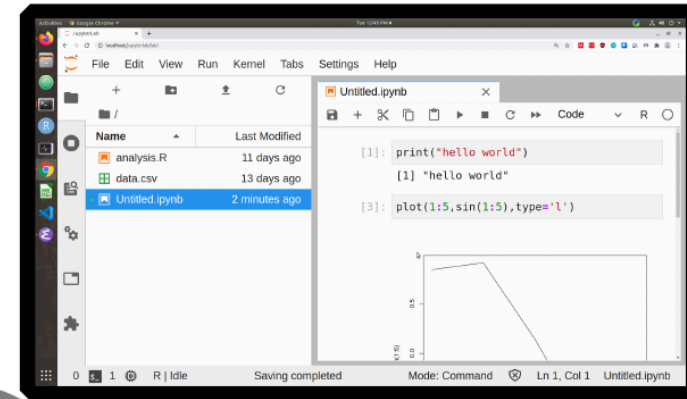
## Original Analysis

Original Computing Environment



## Third Party

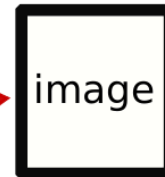
container  
(Copy of Computing Environment)



(1) containerization



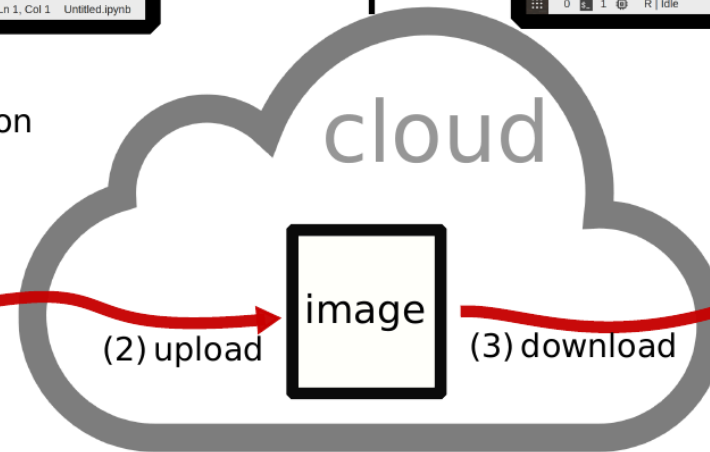
(2) upload



(3) download



(4) run container



# WHAT ARE CONTAINERS

- For a user, feels very similar to virtualization
- But technically, not quite – the kernel is shared
- Very fast – almost native speed
- Still need similar hardware (AMD64 vs M1/2/3 vs ARM etc.)

# WHY USE THEM

- Save (nearly) the **entire computing environment**
  - System libraries and utilities
  - Python, R, etc.
  - Packages
- Fast
- Easy to share
  - Single file
  - Cross platform (Linux, Mac, Windows)

# KEY INGREDIENTS

- Base image
- Dockerfile or Containerfile
- Your existing analysis

# BASE IMAGE

- Minimal Ubuntu
- R (Rocker)
- Jupyter
- Many, many more

# DOCKERFILE

```
(A) A Simple Dockerfile
1 FROM jupyter/datascience-notebook
2 RUN R -e "install.packages('ggplot2', repos =
   'http://cran.us.r-project.org')"
3 COPY --chown=1000 data.csv data.csv
4 COPY --chown=1000 analysis.ipynb analysis.ipynb
5 CMD jupyter lab
```

base image with R and jupyter

R package to install

file on local computer

desired name/location in container

```
(B) Building
1 > docker build -t gjhunt/mwe .
2 ...
3 Successfully built bd5ddab32a75
4 Successfully tagged gjhunt/mwe:latest
```

desired name

# RUNNING

Host Computer Desktop

(A) Terminal

```
1 > docker run -p 127.0.0.1:8888:8888 gjhunt/mwe
2 Unable to find image 'gjhunt/mwe:latest' locally
3 latest: Pulling from gjhunt/mwe
```

-p: ports for browser

image name

(B) web browser

code

files

The screenshot shows a web browser window displaying an RStudio interface. The browser's address bar shows the URL `8888/lab/workspaces/auto-p/tree/analysis.ipynb`. The RStudio interface has a menu bar with 'File', 'Edit', 'View', 'Run', 'Kernel', 'Tabs', 'Settings', and 'Help'. On the left, there is a file explorer showing a directory structure with a 'work' folder and an 'analysis.ipynb' file. Below the file explorer is a table with columns 'Name' and 'Last Modified'. The 'analysis.ipynb' file is highlighted. In the center, there is a code editor with two code cells. The first cell contains the R code `data = read.csv('data.csv')`. The second cell contains the R code `library('ggplot2')` and `ggplot(data=data, mapping=aes(x=Sepal.Length`. Below the code editor is a histogram showing the distribution of 'Sepal.Length' values. The histogram has two bars: a red one on the left and a green one on the right. The y-axis ranges from 10.0 to 12.5. At the bottom of the browser window, there is a status bar showing 'Simple', '0', '\$\_ 1', 'R | Idle', 'Mode: Command', 'Ln 1, Col 1', and 'analysis.ipynb'. Red boxes and arrows highlight the terminal command, the code editor, and the file explorer.

Name	Last Modified
work	5 days ago
analysis.ip...	5 months ago
data.csv	5 months ago

Mode: Command Ln 1, Col 1 analysis.ipynb



# ANOTHER EXAMPLE

(A) Interactive Dockerfile

```
1 FROM johanngb/rep-int
2 WORKDIR /home/rep/
3 COPY data.csv data.csv
4 COPY analysis.ipynb analysis.ipynb
5 RUN jupyter --set-formats ipynb,rmarkdown,R
  analysis.ipynb
6 RUN jupyter nbconvert --to html analysis.ipynb
```

(B) Running an interactive Container

```
1 > docker run -it --name ex_container -p
  127.0.0.1:80:80 gjhunt/mwe:2
2 To get started, please enter the following
  URL into your web browser:
3 http://localhost/
4 ...
```

(C) Welcome Splashpage

Welcome

You may explore the contents of this docker image with any of the following:

- Browse file contents
- Jupyter Notebook
- Jupyter Lab
- R Studio
- Shiny Server

**front end:**  
access from browser  
**backend:**  
running from container

(D) Browse files

**Browse**

Index of /

- ../
- analysis.R
- analysis.Rmd
- analysis.html
- analysis.ipynb
- data.csv

**HTML**

Read in the data in data.csv

```
In [6]: data = read.csv('data.csv')
```

Then we can do some plotting

```
In [8]: library('ggplot2')
ggplot(data=data, mapping=aes(x=
stogram(bins=30))
```

(E) Interact with notebooks

**RStudio**

Console Terminal Jobs

```
~/as: free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and 'citation()' on how to cite R or R packages in publications.
```

**Jupyter**

```
analysis.ipynb
[1]: data = read.csv('data.csv')
Then we can do some plotting
[2]: library('ggplot2')
ggplot(data=data, mapping=aes(x=
stogram(bins=30))
```

# BRIEF TUTORIAL

- Many great tutorials online
- Our paper:

<https://jdssv.org/index.php/jdssv/article/view/53>

# TUTORIAL

I'll be using `podman` which is a reimplementaion of `docker`

Containerfile:

```
FROM jupyter/datascience-notebook
RUN R -e "install.packages('ggplot2', repos='http://cran.us.r-project.org')"
COPY --chown=1000 data.csv data.csv
COPY --chown=1000 analysis.ipynb analysis.ipynb
CMD jupyter lab
```

build

```
podman build -t ex_image .
```

run

```
podman run -it --rm -p 8888:8888 ex_image
```

# TUTORIAL

saving

```
podman save ex_image -o ex.tar
```

loading

```
podman load -i ex.tar
```

# TUTORIAL

listing images

```
podman image ls -a
```

listing containers

```
podman container ls -a
```

# **COMMENTS AND DISCUSSION**

# ADDITIONAL ADVANTAGES OF CONTAINERS

Our goals:

1. Exactly reproducible
2. User friendly
3. Transparent
4. Reusable
5. Archived
6. Version controlled

# USER FRIENDLY

- Code easy to access and inspect, ideally even without downloading
- Should require minimal effort for a user to install and run
- Should cause minimal disruption to a user's resources (e.g., not install unwanted software on their system)
- etc.
- Minimize the user's security concerns



# DISCUSSION