PIE Lab Manual

The PIE lab

2020-07-05

Contents

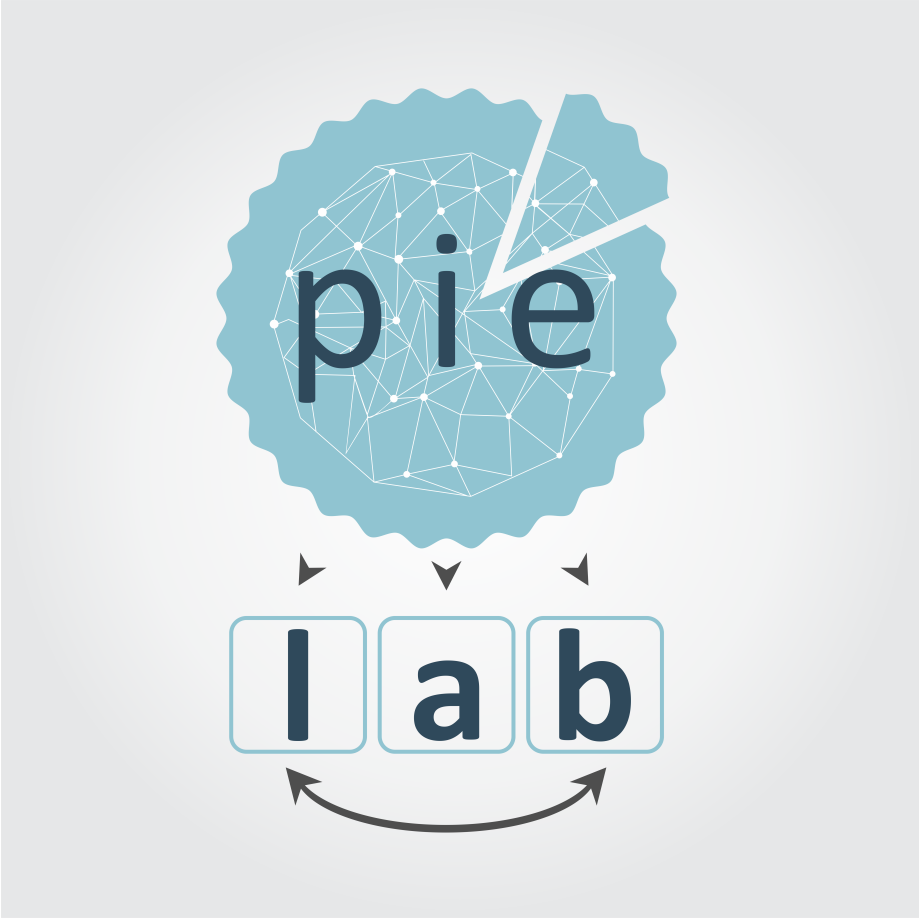

Figure 1:

Chapter 1

About the lab

1.1 Location

PIE lab – Straub 417 and 419

1.2 People

1.2.1 PIs

David M. Condon

- Office: Straub 323
- Email: dcondon@uoregon.edu
- Twitter: @DMCpersonality

Sara J. Weston

- Office: Straub 325
- Email: sweston2@uoregon.edu
- Twitter: @sarawestonO9

1.3 Values

1.3.1 Transparency

Science must be verifiable by others. For this reason, we strive to be transparent in our work. We share code, materials and, when possible, data. We will admit

mistakes.

1.3.2 Curiosity

1.4 Expectations

1.4.1 Honesty

1.4.2 Respect

1.5 Network Drive

Files can be shared among lab members using the lab network drive.

1.5.1 Accessing the drive on a Mac

1. Go to 'Connect to Server'.
2. If you're off-campus, connect to the UO VPN.
3. Enter the network address: `smb://cas-fs2.uoregon.edu/Psychology/a/PIElab`. (It's worth adding this as a favorite network so you don't have to remember the address again.)
4. Enter your DuckID and your password when prompted. Note that you can only gain access if you have been approved for access.

1.5.2 Accessing the drive on a PC

1. Follow directions here.

Chapter 2

Science Resources

2.1 UO resources

2.1.1 Virtual Private Network (VPN)

Instructions are here.

2.2 R

2.2.1 Learning R

- Learning Statistics with R by Danielle Navarro
- R for Data Science by Hadley Wickham

2.2.2 Reference guides

- Cheatsheets

2.3 Writing

- Dan Simons's writing guide
- E.J. Wagenmaker's guide

2.4 Presentations

- Presentation by Rachael Meager

Chapter 3

Talapas Overview

Talapas is the high performance computing cluster managed by Research Advanced Computing Services (aka RACS) at the University of Oregon. Since it is possible (and maybe even best) to get started with Talapas without knowing too much about the capabilities of the system, we will simply point to the main website and the Talapas Knowledge Base for learning more.

In this document, the numbered sections below will deal with separate tasks that users in the lab might want to perform on Talapas. If you learn how to do something that is not documented below, please add it to the list, even if you are unsure that others will find it useful. Or, just point to the documentation available online.

3.1 Requesting Access

Go to this website to request access to the PIE lab Talapas account. Fill out the New Account Request Form, including **pielab** as the PIRG.

3.2 Using Open OnDemand Interactive Access

The RACS team recommends using an incognito tab in Chrome or Firefox before logging in to Talapas (Safari will not work), as this may help to limit the potential for someone to misuse your credentials. Bear in mind that computing services can be costly so there is more at risk than your personal info (and data).

In the browser window, go to: https://talapas-ln1.uoregon.edu

Enter your username and password. For UO users, your username on Talapas will be your Duck ID. That is, if your email address is alice@uoregon.edu, your

Talapas username will be "alice". Your password is the same university-wide, and can be managed at the UO password reset page. If you don't know your credentials (or if you're not sure you have credentials), visit this page to request access. This will require the approval of a lab PI (Condon or Weston). Since there are fees associated with using the computing cluster, your request will be evaluated based on need and the availability of other options.

There is a brief overview of the OnDemand service worth reading — see here.

3.2.1 Loading files for Interactive Access

Once you have logged in, it's fairly intuitive to navigate the dropdown menus at the top of the browser window. The 'Files' dropdown gives several options and each will open a new window. For shared projects, consider using the '/projects/pielab/shared' directory to store your files; otherwise use the directory associated with your username *within the pielab directory* (in other words, this one: '/projects/pielab/[username]'). Within the correct folder, you will want to create a new folder that is specific to each project as this is where you'll store the input and output files, just as you would if working on your own hard drive.

3.2.2 To launch an interactive session

From the OnDemand landing page, click 'Interactive Apps'/'Talapas Desktop'. This will load a form that requires entry of the PIRG name (pielab) and a few other fields. There are several things to keep in mind when filling out the form to launch the virtual desktop on Talapas. First, the 'partition' field requires selection of one of several options. For a complete list, with associated specs, limitations, and restrictions, see here. For many (most?) jobs, the default 'interactive' partition should be fine. For more intense jobs, consider using 'short' (gives up to 24 hours) or 'long' (up to 2 weeks). Note that 'interactive' has a max of 4 CPUs, which equates to about 16G RAM. For reference, a souped up laptop might have 32G and a standard MacBook Pro has 16G or maybe 8G if older. Analyses of SAPA data from 2017 or later will routinely kill RStudio with only 16G of RAM (necessitating use of the 'short' or 'long' partitions).

For the number of hours, a good practice is to use the lesser of the max amount of time (this depends on the partition) or a generous guess at the length of time needed. If the analysis runs for a long while and then times out, you will have no output and will have spent the funds anyway. Better to have a chance of getting what you need than running out of time. However, you should also get into the practice of deleting instances that are no longer in use, as this will save money! Otherwise, the expenses will add up until the full time allotment has expired. For the number of cores and total memory, the standard seems to be 4G of memory for each CPU though some partitions (fat and longfat) have CPUs with more memory — choose the CPUs accordingly (as best you can).

Note that you can also use GPUs, which seem to have more memory (and cost more). See here for more info about guessing the right specifications.

For SAPA processing, I have done lots of experimenting. Unfortunately, most of the analyses conducted on these data are not able to be parallelized, and this includes the most computationally expensive step of getting the count of pairwise administrations across all items. This means the marginal benefit of invoking additional CPUs is often small (but not zero). When going from raw data to a version that is Dataverse-ready, this set-up seems to work pretty well: 0 GPU, 200 GB, 14 cores, on the preempt partition for at least 24 hours.

Note that if you seek to use more than 128 GB (as above), your request will end up being completed on the GPU partition (which has 256 GB per node) or the fat/longfat/preempt partitions (which can handle up to 200 GB, per my correspondence with the RACS team). As of July 2019, the gpu partition costs twice as much as the short partition ($0.008 per SU) and the fat partition costs 6x as much. I'm not sure how much the preempt partition charges but it's probably in the same range. The preempt partition is nice because it will typically start more quickly than if you use one of the others, though you run the risk of having the job killed if traffic on Talapas picks up among higher-priority users. See here for more info. This won't happen with fat or longfat, but you might have to wait a while before your job can start.

3.2.3 To determine the memory usage of jobs previously run

This has to be done through an ssh interface with Talapas. Open a shell and enter this at the prompt:

```
ssh username@talapas-ln1.uoregon.edu
```

followed by your password.

To see the memory usage of recent jobs (since midnight of the current day), enter:

```
sacct --format='JobID,Elapsed,MaxRSS'
```

To get more information on an individual job (recommended), enter:

```
seff <JobID>
```

That last command will return something like:

```
$ seff 9609990

Job ID: 9609990
Cluster: mycluster
User/Group: username/talapas
```

```
State: CANCELLED (exit code 0)
Nodes: 1
Cores per node: 8
CPU Utilized: 00:23:25
CPU Efficiency: 3.10% of 12:36:16 core-walltime
Job Wall-clock time: 01:34:32
Memory Utilized: 29.12 GB
Memory Efficiency: 91.00% of 32.00 GB
```

This job was cancelled after repeated efforts to compile. Note the high ratio of Memory Utilized to Memory Available. It is also possible to learn much more. Try using the code below by entering your own 'username' and a date that goes back a few days in place of 'year-mo-da'.

```
sacct -u <username> -S <year-mo-da> --format='JobID,Partition,State,AllocCPUS,ReqMem,Ma
```

This gives all of the most relevant info (in my opinion) on jobs from the date entered until now, including the final status of the job.

To check on memory usage while a job is running, you can enter 'top' in the terminal window to see info about the CPUs currently in usage. Note that you will see slightly different info if you do this through a remote login to the terminal window vs opening a terminal window within the 'live interface' (see below for more about this option).

Finally, it is also possible to obtain information about specific files. This is useful if trying to determine how much time was needed to generate an output file. From the terminal, try a command like this:

```
ls -l ~/projects/<filename>
```

This will give the size of the file along with the date and time of creation.

3.2.4 To run a live session of RStudio

Once you have begun the session (i.e., submitted the form), a new page will load. From here, click the blue button that says 'Launch noVNC in New Tab'. This button may not be visible right away as you sit in the queue, waiting for the necessary resources to become available. When you launch the tab, a virtual desktop will open with a dropdown menu in the upper left called 'Applications'. Choose 'Education'/'Talapas RStudio'. Presumably you can figure out how to manage your workflow from here. You can also open a terminal window through this interface that will allow you to access the node you're running. To do this, click on the terminal icon in the 'TurboVNC' tab — it's to the right of the 'Applications' dropdown menu.

3.2.5 Ending your interactive session

Your analyses will continue running until completed, timed-out, or stopped for some other reason (coding errors, etc.). When your analyses are done, you should return to the Open OnDemand main page and click the red 'Delete' button. This will stop your usage of the system and associated fees.

3.3 Using Talapas for Batch Jobs

I haven't done this yet (on Talapas)! But see here if you want to give it a go.

Chapter 4

Onboarding

4.1 CITI training

All lab members must have appropriate CITI training in order to be included on IRB protocols and grant applications.

4.1.1 Where to get CITI training

Click on this link in order to access the CITI Single Sign On (SSO) platform.

If you're a new user log in using your DuckID and follow the instructions to create an account.

4.1.2 Required courses

Everyone - Protection of Human Research Subjects

For Grant Applications - Responsible Conduct of Research

4.1.3 When you're done

Save a copy of your certificates somwhere you can easily access them.

Chapter 5

Website and Lab Manual

5.1 Website

5.1.1 Details

URL: pielab-science.com

Username and password: email Sara or David for this information.

5.1.2 Purpose

PR for the PIE lab - Who's part of the PIE lab - What are we currently doing? - (If actively recruiting) landing page for new recuirts - (If ongoing study) landing page for study participants

5.1.3 Current sections

- What we study
- About
- Current lab members (with a page for each)
- Friends of the lab
- Cooling on the rack (news)

5.1.4 How to update

- Go to pielab-science.com/admin and log in (see above for username/password details)

- See sections below for updating specific aspects

5.1.4.1 Your profile

- Go to the Media section (on the left-hand menu) and click Add New
- Drop file into box, or select using menus
- Go to Pages (left-hand menu) and hover over your page – select Edit with Elementor
- Use the interface to change text, upload a new picture, include links to other pages, whatever you want!
- Be sure to click Update before leaving the page.

5.1.4.2 Your CV

- Go to the Media section (on the left-hand menu) and click Add New
- Drop file into box, or select using menus
- Go to Pages (left-hand menu) and hover over your page – select Edit with Elementor
- Hover over the box that contains the text. A pencil will show up in the corner; click on it.
- On the left-hand side, an Edit Text Editor menu will show up. Find the link to your CV. If you click on it, the link to the PDF will appear. You can either
  - Edit the link to match the new upload (involves changing the year and month of upload and the name of the file as it was uploaded to Wordpress), or
  - Delete the current link to the CV and add a new one with the Add Media button.
- Be sure to click Update before leaving the page.

5.1.4.3 Troubleshooting

If the page doesn't update

- Wait 15 minutes and reload the webpage.
- Clear the wordpress Cache
  - On the wordpress dashboard, click on Mangaged Wordpress (top) and select Flush Cache. This will take a minute or so.
- Clear your browser Cache
  - This will be under your browser's settings.
  - Try opening the webpage in an incognito window as well, to see if this is a problem.

5.2 Lab Manual

5.2.1 Clone lab manual to your computer

Notes:

1. *this is only available to current graduate students and PIs in the lab.*
2. *the steps listed here only need to be followed on the first use.*

Necessary materials:

- Git
- GitHub account with access to PIElab organization
- Rstudio that is connected to GitHub through your personal account.

1. Go to manual repository.
2. Click on the green *Clone or download* button.
3. Copy the link that appears.
4. Open RStudio. Create a New Project.
5. Select Version Control and then Git. Name the project "manual" (so it matches the Rproj file that already exists in the repository. Save this project somewhere you will remember.

5.2.2 Updating the manual [using the RStudio GUI]

Before you make ANY changes:

1. Open the manual RStudio project.
2. Pull the most recent version of the manual using the blue down arrow.

Make any changes to the manual as you think are necessary.

3. In the Build tab in Rstudio, click on *Build Book*.
4. In the Git tab, click Commit. In the commit message section, briefly (4-5 words max) describe the changes you made. Example: wrote updating manual section. In the window on the left, the easiest thing to do is select everything and click *Stage*. Ensure all the boxes are checked. This will be slower than only selecting the parts of the book that have changed. However, the nature of an RMarkdown book is many changes affect most parts of the book (because it changes you navigate to them). If you miss a change you have made, the book may not render properly online. Click the *Commit Button*.
5. Click on the green up arrow to push the book to GitHub, which will update the book online.

5.2.3 Updating the manual [using the terminal in Rstudio]

Before you make ANY changes:

1. Open the manual RStudio project.
2. In the terminal window of RStudio, pull the most recent version using this code:

```
git pull
```

Make any changes to the manual as you think are necessary.

3. In the Build tab in Rstudio, click on *Build Book*.
4. In the Terminal, commit your changes. The easiest way to do this is to type:

```
git add .
```

where the period indicate to include anything. Alternatively, you can add only specific files, if you don't want to commit everything. For example:

```
git add file.R
```

or:

```
git add path/file.R
```

To commit your changes, type:

```
git commit -m "message here"
```

Be sure to include a message describing the changes. Keep the message short.

Finally upload your changes with:

```
git push
```

You're done!

5.2.4 Requesting updates

You may be unable to make changes to the lab manual because you are not authorized to make changes or don't know the answer to the question. In either of those cases, you make create an issue on the GitHub page to alert the lab to the need for additional material. To do so:

1. Go to the Issues tab on the GitHub page for the lab manual.
2. Click on New Issue and complete the form. Be sure to include as much detail as you can about the issue you have. The more detailed your request, the better the information in the manual will be.

Note: create one issue request for each question you have. Don't put lots of requests into a single issue.

Other lab members can take responsibility for this issue by self-assigning the issue. Once material has been added to the manual, the member responsible for resolving the issue should reply to the original poster, including a link to the relevant section. If this meets the need of the original poster, they should reply so and the issue can be marked as closed.

Chapter 6

Life

6.1 Taxes

Note that being a GE and having a research fellowship are considered different forms of compensation. Here's a guide to preparing your (2019) federal taxes.

Chapter 7

SAPA project website

7.1 Introduction

The SAPA Project hosts surveys that explore different dimensions of personality.

7.1.1 Infrastructure

The SAPA Project is hosted on Google Cloud. There is one active compute engine virtual machine (VM) instance, and one backup or staging instance. The VM instances run an Apache/2.4.25 web server on Debian 9 (stretch) Linux based operating system that hosts the actual SAPA Project website. Each time a user completes a survey, the survey results are stored in a MySQL database, which is running on a Google Cloud SQL instance running MySQL 5.7.

7.1.1.1 Admin console

Admin console for the computing infrastructure is available at https://console.cloud.google.com.

7.1.1.2 How to add new administrators / owners

Add a new user from the IAM & Admin admin console page.

7.1.2 How to connect to database

First time users:

1. Download and install the Google Cloud SDK
2. Download and install the Google Cloud SQL Proxy
3. Download and install a MySQL client. On macOS, use homebrew to install a command line client via `brew install mysql`, or you can install the MySQL Workbench GUI client

- In a terminal session, authenticate to Google services

```
gcloud auth login
```

A web browser will open and direct you to login to your Google account.

- Start the `cloud_sql_proxy`:

```
cloud_sql_proxy -instances=silken-alloy-248920:us-west1:woodworth-pi-improvement-proje
```

If the command succeeds, you'll see the following message returned to your shell:

```
Ready for new connections
```

The Cloud SQL Proxy allows you to make connections to the MySQL database running in Google Cloud as if the database was running locally.

- In a new terminal window, verify the connection:

```
mysql -h 127.0.0.1 -u SAPAreader -p
```

Enter the password when prompted. If the connection is successful, then you'll see the mysql prompt:

```
mysql>
```

7.1.3 Connecting from R

Following the instructions here: https://cloud.google.com/blog/products/gcp/google-cloud-platform-for-data-scientists-using-r-with-google-cloud-sql-for-mysql

It is helpful to create a function that creates the connection to the MySQL database:

```
library(RMySQL)

getSqlConnection <- function(){
  con <-
    dbConnect(
      RMySQL::MySQL(),
      username = 'SAPAreader',
      password = 'password',
      host = '127.0.0.1',
      dbname = 'SAPAactive'
```

```
    )
  return(con)
}
```

Then you can query any table in the `SAPAactive` database by following this example, which queries 10 results from the `TAIE_responses_111119` table:

```
conn <- getSqlConnection()
res <- dbSendQuery(conn, "select * from TAIE_responses_111119 limit 10")
data <- dbFetch(res)
print(data)
dbDisconnect(conn)
```