Center for Conservation Biology | UC Riverside

Research Unit Resource Guide

Lynn Sweet | Principal Investigator, Asst. Research Ecologist
Julia Parish | Data Science Fellow

2022-10-04

Contents

Welcome to the CCB Research Unit Resource Guide

The Center for Conservation Biology (the Center, or CCB) is a University of California Riverside (UCR) organized research unit. Our mission is to assist in the conservation and restoration of species and ecosystems by facilitating the collection, evaluation, and dissemination of scientific information.

East of Cima Dome in the Mojave National Preserve, California. **Image Credit:** Center for Conservation Biology


UNIVERSITY OF CALIFORNIA
UCRIVERSIDE
Center for
Conservation
Biology
UC Riverside, Palm Desert

Chapter 1

Introduction

This Resource Guide provides an overview of the the UC Riverside Center for Conservation Biology (CCB) research team work processes and expectations. It hosts documentation on how to install software and project management tools utilized by the CCB team to facilitate collaborative research.

If you have suggestions for additions or changes, please make a pull request or contact Lynn Sweet (lynn.sweet AT ucr.edu).

1.1 Computer requirements

CCB work computers (laptop or desktop) operating systems should either be Windows or macOS. Please note that Windows based machine it required to run ESRI ArcDesktop and ArcPro software. Both Windows and macOS may utilize ESRI ArcOnline tools.

For **Mac** users, update macOS to the newest supported version. Navigate to System Preferences –> Software Update.

For **PC** users, ensure you have Windows 10 or 11 installed. If not, request a Windows key from UCR IT at UC Riverside ServiceLink

1.2 Software Installation

Software installation covered in this resource guide includes:

- ESRI Platform
  - ArcGIS Desktop

  - ArcGIS Pro
  - ArcGIS Online
- git
- GitHub
- Google Apps
- R
- RStudio
  - Quarto
  - Bookdown
- Slack
- Trello
- Zotero

Thank you to UC Santa Barbara's Bren School of Environmental Science & Management and National Center for Ecological Analysis and Synthesis (NCEAS) staff for providing many of the resources listed in this reference guide. Information was made available on the UCSB-MEDS GitHub page.

Chapter 2

R and RStudio Installation

2.1 Install R

R is a programming language and environment used for statistical computing and grahics. For more information, please visit What is R.

To install R, visit cloud.r-project.org to download the most recent version for your operating system.

Would you like a visual guide for install R and R Studio? Check out this YouTube R for Ecologist

2.1.1 Updating R

If you already have R installed, you may want to update the version of R you are using. Usually this occurs with a major version upgrades versus minor updates.

Please be aware that when you update R, it means re-installing all of the packages you have previously installed. It is worth updating R when major updates are released, especially if new packages you are attempting to install are not compatible with the version of R currently installed.

In order to update R, you have to find your installed version of R and run it on its own, outside of RStudio. This is easy if you have an R desktop shortcut, but not too hard if you hunt around a bit in your Applications folder.

Be Mindful

Do not just upgrade R to a new minor (or major) version in the middle of a project. This will take a bit of work and time, maybe an hour if you have a lot of packages installed, as this will require re-installing packages to a new library folder.

Please see details below in Section 2.1.2 to save all R packages currently installed PRIOR to updating R, so you won't have to reinstall them individually after the update

Double click the R icon to start up R. It will open the R Console and a menu. Click on the R menu at top left, and select Check for R Updates. If you are up to date, the R Console will report "Your version of R is up to date". If not, this process will provide windows and buttons to click to upgrade to the latest version of R. When done, quit R and start RStudio to make sure the update has carried over. You should see the new version number when RStudio starts.

2.1.2 Bulk Save R Packages Prior to Updates

In order to follow the bulk save R package instructions, you will need to have R Studio installed, and be familiar with the terminal and console panels.**

Before you jump into upgrading R versions and packages, make sure that any important projects are protected. Make sure that your important ongoing projects use the {renv} package to preserve their Environment, and that you have taken a recent snapshot with `renv::snapshot()`

Before updating R, check out where your current package library is saved by entering `.libPaths()` in the console of RStudio. This should look something like:
`[1] "/Library/Frameworks/R.framework/Versions/4.2/Resources/library"`

Note that this package library is specific to the most recent major and minor version of R. If you upgrade from 4.1.x to 4.2.x, you will have a new library path to a new 4.2 folder, and will need to re-install packages in this folder.

You can do this from scratch, and install packages individually as needed (which is not a bad way to clean out packages you are not using), but you may prefer

to re-install all packages in an organized way, rather than when you need to use one

You can see a dataframe of all currently installed packages by running the base R function `installed.packages()` in the R Studio console. This will help make a list of packages prior to updating R.

There is also a package for **Windows**, {installr}, which will pull the list of your current packages and install these in the correct folder for the new version of R. If you are on a Windows OS, and have installed the {installr} package, the `installr::updateR()` command in the console performs the following:

- finding the latest R version
- downloading it
- running the installer
- deleting the installation file
- copy old packages to the new R installation
- updating old packages as needed for the new R installation

But for MacOS and UnixOS folks, this must be done manually.

2.1.2.1 Saving a List of R Packages Manually

The short script below will save a vector of your current packages to a file in your root (home) directory, named `installed.packages.rda`. Copy this code chunk below into your Console pane in RStudio and run it. Then reopen RStudio, go to the Files tab, click on Home , and check your root (home) directory for the installed.packages.rda file. You can click on this file to load it into your Environment tab (answer yes to load). In your Environment tab, it should look like a character vector with the names of all of your packages in quotes.

```
# save list to tmp as a matrix object
tmp = installed.packages()

# filter for all rows where the 'Priority' column is NA,
# and select just (column 1) the package names in quotes,
# and then assign this vector to the installedpackages
# object
installedpackages = as.vector(tmp[is.na(tmp[, "Priority"]), 1])

# save this vector as an *.rda file in your root (home)
# directory
save(installedpackages, file = "~/installed_packages.rda")

# remove the tmp matrix and the installedpackages vector
```

```
# from your Environment
rm(tmp)
rm(installedpackages)
```

Now that you have your saved list of packages stored in a file, you are ready to upgrade R. Quit RStudio, and go to r-project.org.

2.1.2.2 Reinstalling list of R Packages

First, let's check your new R version with `R.Version()` - run this in the Console pane. This should be the new version.

Now let's check your library path with `.libPaths()` - run this in the Console Pane - this should match your R version.

Now run `installed.packages()`. This should show that your new library is mostly empty, except for the base R packages. We will now fix that.

Now you need to take advantage of your nicely-stored list of packages to reinstall all of your packages in your new (mostly empty) library.

Use the short script below to load the file `installed.packages.rda` to open a vector of your previous packages in your Environment pane, and then re-install all of these packages. Copy the code chunk below into your Console pane in RStudio and run it. This will take a while, especially if you have a lot of packages installed.

While you are waiting, check out the code chunk below. The first step is to load the installedpackages.rda file from your root (home) directory into your Environment pane. Then it starts taking action on this vector of package names. See what it does in the next step, the for loop, and think about how it works. - It measures the length of the vector, installedpackages. - Then it counts from 1 to the length of installedpackages. - For each count, it installs the package at position [count] in the vector - Then it goes to the next [count] value, and installs the next package in the vector - Until it reaches the full length of the vector (installs the last package), and then stops.

```
# loads vector into your Environment pane
load("~/installed_packages.rda")

# For each package in this vector of length [count],
# install them one by one until the last one is installed.

for (count in 1:length(installedpackages)) {
    install.packages(installedpackages[count])
}
```

When all of the installation is complete, you can check your work by running `installed.packages()` again in the R Studio console. You should see a full list of your packages, and you are updated and ready to go!

2.2 Install or Update R Studio

RStudio is a software (considered an Integrated Development Environment, or IDE) that provides R programmers with an easy-to-use interface for coding in R.

Note: RStudio will not work without R installed, and you won't particularly enjoy using R without having RStudio installed. Be sure to install both!

- **New install:** To install RStudio, visit rstudio.com/products/rstudio/. Download the free ("Open Source Edition") Desktop version for your operating system.
- **Update:** If you already have RStudio and need to update: Open RStudio, and under 'Help' in the top menu, choose 'Check for updates.' If you have the most recent release, it will return 'No update available. You are running the most recent version of RStudio.' Otherwise, you should follow the instructions to install an updated version.

Open RStudio (logo you'll click on shown below). **If you are prompted to install Command Line Tools, do it.**

Mac Users

There may be a need to install command line tools and XQuartz:

- To install command line tools (if you're not automatically prompted), run in the Terminal tab in RStudio: `xcode-select --install`
- Visit xquartz.org to download & install XQuartz.

2.2.1 Update R Packages in R Studio

You will want to periodically update R packages to ensure you have the most up to date version. You can do this within RStudio by navigating via the Task Bar to `Tools/Check for Package Updates`.

- Select All, and let the updates begin.

2.3 Install Quarto

This is an optional tool within R Studio that is extremely powerful, but it is not required.

Quarto is a scientific publishing tool built on Pandoc that allows R, Python, Julia, and ObservableJS users to create dynamic documents, websites, books and more.

As of *July 2022*, Quarto comes pre-installed in R Studio (v2022.07). If you haven't updated your R Studio IDE (and concerned about doing so), you can install Quarto separately.

- Download Quarto here and install
- To use Quarto through the RStudio IDE, be sure to have at least version v2022.02 installed (see directions in step 2, above)

Learn more about Quarto here.

2.4 Learn How to Use R & RStudio

There are many resources out there to help you learn how to use R and the RStudio IDE (YouTube, Googling, StackOverflow, etc…). This is a short list of primary resources for you to reference. We encourage you to join R community Slack workspaces, R User community groups (such as R-Ladies), and UC Riverside Data Science clubs!

- ACM at UCR - UC Riverside community dedicated to technical, professional, and personal development in the context of computer science.

- R for Data Science
- R Markdown Cookbook
- Cookbook for R
- Advanced R
- R Studio (posit) Book Catalog
- R for Data Science Online Learning Community
- R OpenSci
- RStudio Cloud - interactive tutorials to learn data science basics.
- R Studio Cheatsheets - invaluable tool to learn how to use various R packages.
- Swirl - R package that is a built in tutorial.
- Teacups, Giraffes, & Statistics - an interactive tutorial to learn statistics and R coding, plus it is a beautiful!
- Data Science in a Box
- Adventures in R
- Data Analysis & Visualization in R for Ecologist

Chapter 3

Bookdown Guide

This Resource Guide was created using the `bookdown` package. **bookdown** provides a platform to publish references using various output formats (PDF, HTML, Word) and allows for interactivity, insertion of code chunks, and supports a variety of languages (R, Python, Julia, SQL, etc.).

3.1 Primary Reference Resources

Here is a list of resources to learn how to use and edit in bookdown

- Bookdown Package Documentation
- Authoring Books with R Markdown
- R Markdown Cookbook
- R Markdown: The Definitive Guide

3.2 Install the Bookdown package

The first step to edit and add to this Resource Guide is to install the **bookdown** package from CRAN or Github. In the RStudio console, run the following:

```
install.packages("bookdown")
# or the development version
# devtools::install_github('rstudio/bookdown')
```

3.2.1 Additional packages

You may need to install additional packages if you do not currently have them in order to render the Resource Guide bookdown:

- `downlit`
- `formatR`
- `here`
- `kableExtra`
- `Rtools`
- `shiny`
- `tidyverse`

3.2.2 Install TinyTex

TinyTeX is a custom LaTeX distribution. LaTeX, which is pronounced «Lah-tech» or «Lay-tech», is a document preparation system for high-quality typesetting. It is most often used for technical or scientific documents. To learn more about TinyTex, navigate to this document, and to learn more about LaTex, visit the LaTex Project site.

To install TinyTex on your computer, navigate to the R Studio console panel, and type in the following command:

```
tinytex::install_tinytex()
```

3.3 Render, Build & Publish the Resource Guide

When updates are made to the Resource Guide, it needs to be rendered, built, and then the updates published.

In your Console, type either of these commands depending on which type of render you prefer:

```
bookdown::render_book("index.Rmd", "bookdown::gitbook") bookdown::render_book("index.Rm
"bookdown::pdf_book")
```

Note: If there is an error during rendering, Google what the error message means! You should be able to find an answer on stack overflow or other tech support site. One of the most common errors during rendering is that not all dependencies (aka packages) have been installed on your local computer.

Once you have rendered updates using the console commands above:

- navigate to the **Environment, Connections, Build, Git** panel,

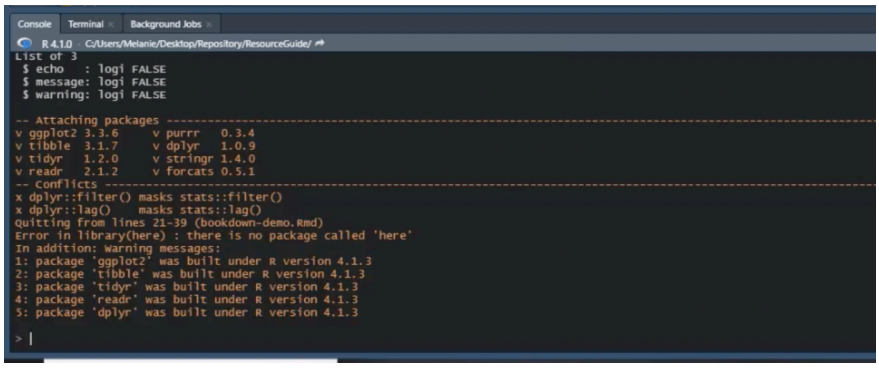
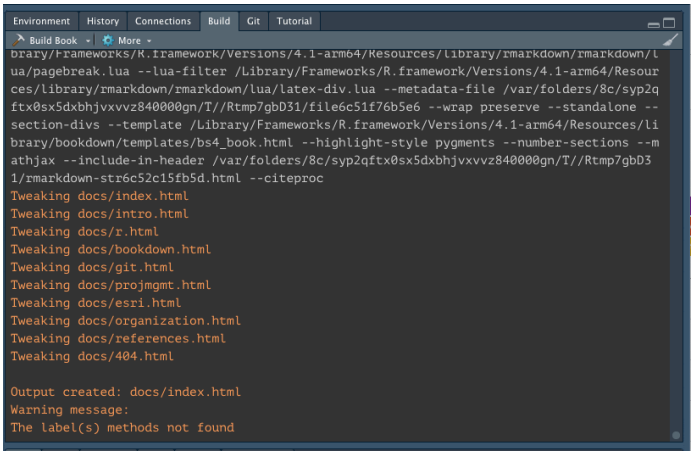
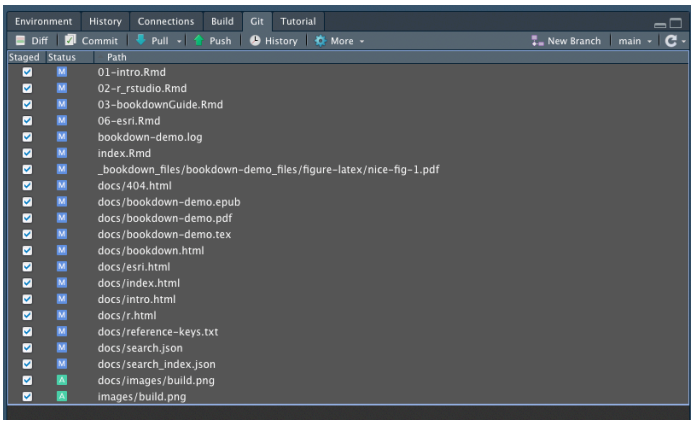
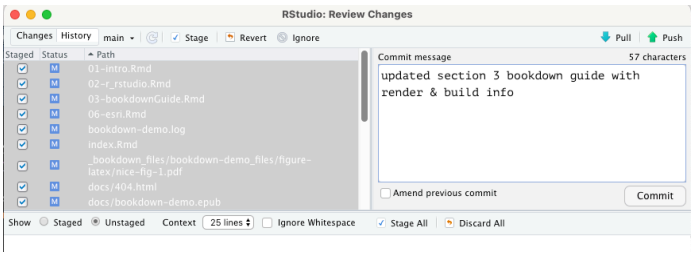


Figure 3.1: An common error message in the RStudio console: Error in library(here): there is no package called 'here'. If you see this library error message, all you need to do is install the required package! The additional warning messages are notifications that R should be updated to be compatiable with the current packages. These will not hender the render.

- select the **Build** tab,
- select **Build Bookdown**
- once the build is complete (the final line should read `Output created: docs/index/html`), commit, pull and push updates to GitHub via the **Git** tab.







The following information is directly taken from the *bookdown* package (Xie, 2022).

3.4 Formatting

You can use anything that Pandoc's Markdown supports, e.g., a math equation $a^2 + b^2 = c^2$.

3.4.1 Referencing Bookdown Chapters

Remember each Rmd file contains one and only one chapter, and **a chapter** is defined by the first-level heading `#`.

You can label chapter and section titles using `{#label}` after them, e.g., reference this Resource Guide Chapter 1 by using the markdown format: `\@ref(insert chapter label name here)`. If you do not manually label them, there will be automatic labels anyway, e.g., Chapter **??**.

3.4.2 Formatting Figures

Figures and tables with captions will be placed in `figure` and `table` environments, respectively.

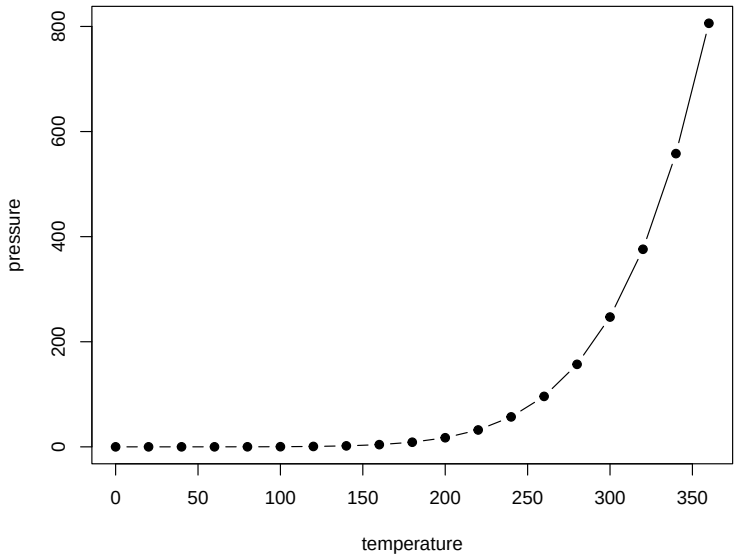


Figure 3.2: Here is a nice figure!

Reference a figure by its code chunk label with the `fig:` prefix, e.g., see Figure 3.2. Similarly, you can reference tables generated from `knitr::kable()`, e.g., see Table 3.1.

3.4.3 Citations

You can easily write citations using .bib files within this repository formatted using BibTEX. For example, the **bookdown** package (Xie, 2022) in this reference book, which was built on top of R Markdown and **knitr** (Xie, 2015).

3.4.4 Alt Text for Accessibility

Use the knitr package to add alt text to graphics in R Markdown files

3.4.5 Migrating to Quarto from Bookdown

Love `bookdown`? Check out `Quarto`! To learn how to convert `bookdown` documents to `Quarto`, check out this blogpost on Openscapes

Table 3.1: Here is a nice table!

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3.0	1.4	0.1	setosa
4.3	3.0	1.1	0.1	setosa
5.8	4.0	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa

Chapter 4

Git Installation & GitHub Account

4.1 GitHub

GitHub is a internet based code hosting platform for collaboration and version control. GitHub lets you (and others) work together on projects.

Navigate to github.com, and create an account! Please use either your work or personal email account. You can add several emails to your account, and assign a particular email as the primary email for the account.

Review this article on choosing a GitHub username: happygitwithr.com/github-acct.html.

Want a cool Octocat icon for your GitHub account? Check out these icons: - Octodex - Build Your Own Octocat

Once you have created a GitHub account, join the CCB's GitHub Organization. Please email the CCB PI to request they invite you to the organization.

4.2 Git

Git is a tool utilized for source code management. It tracks changes and updates to code and allows for multiple users to work together within the same repository. Recommended reading for background information on Git and its utility (Yıldırım, 2020).

Check to see if your computer already has git:

1. Open RStudio
2. In the terminal, run the following command:

- `which git` (Mac, Linux)
- `where git` (Windows)

If `git` is already installed, the return to the above command should return a filepath similar to:

- MacOS `/usr/local/bin/git`
- Windows `C:/Program Files/Git/bin/git.exe`

3. If there is no response, download and install `git` from here: git-scm.com/downloads. Select the default settings within the prompts **except** the default to **master** branch. This branch is being phased out, so select the option that let's you select alternative branches (ex: main).

Once Git is installed and your GitHub account has been set up, Git needs to be configured on the computer. Configuring Git is required to push & pull commits to GitHub.

Configuring Git:

1. In RStudio, open the terminal.
2. Run the following commands separately, pressing `Enter` after each line. Replace username and email with your GitHub account username and email. Make sure to keep the quotes around the username in the code below:

- `git config --global user.name "Jane Doe"`
- `git config --global user.email janedoe@example.com`

3. Once you have entered the above command line code, check that the configuration was set correctly. In the RStudio Terminal, type the following command and hit enter:

- `git config --list --global`

In the terminal, it should show code similar to this in the Terminal:

`user.name=janedoe user.email=janedoe@ucr.edu core.excludesfile=/Users/janedoe/.gitignore`

If when installing or updating Git, the default branch was not set to `main` (it is defaulting to the old `master` branch), you can change this setting globally. In the Terminal again, enter the line below:

```
git config --global init.defaultBranch main
```

For more information on configuring Git: check out this Git reference

4.2.1 Personal Access Token

Once Git has been configured to commit to your GitHub account, a **Personal Access Token (PAT)** must be created for **each computer you intend to use**. A PAT is an alternative password authentication method for Git to access GitHub accounts.

- In the RStudio Console, install the `usethis` package in R by running the following code:

```
install.packages("usethis")
```

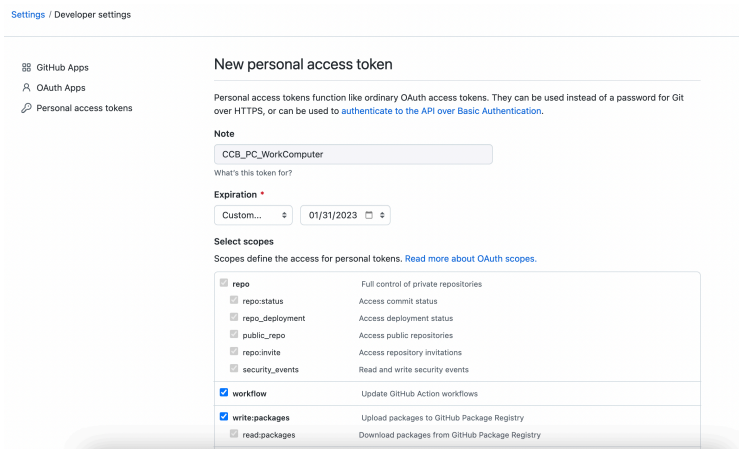
If the `usethis` package is installed correctly, at the end of the stream of text there should be a message similar to the image below:

- Once the `usethis` package is installed, run the following in the RStudio Console:

```
usethis::create_github_token()
```

- Enter your GitHub password when prompted.

This should take you to the **Settings/Developer settings** section of your GitHub account:



- **Note Field**

Change the PAT name to a meaningful reference (see image for an example). You may end up creating multiple PATs, so you want to ensure that you know which PAT is designated for each computer | server.

- **Expiration Field**

This is to select a set expiration timeframe for your PAT. Setting an expiration is highly recommended, and GitHub will send you *SEVERAL* emails prior to it expiring to remind you to renew it. Use the drop down to select a set time frame (7 days to 90 days) or create a custom expiration time frame (exactly a year or particular date).

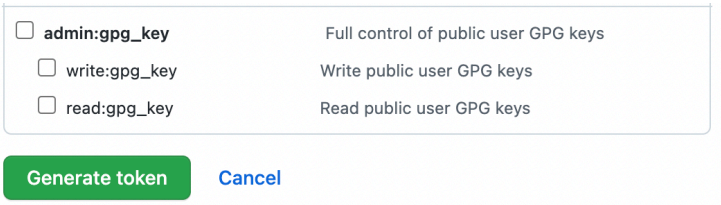
- **Select Scopes Field**

Define access for the Personal Access Token being generated.

It is recommended to select at least the following scopes: - repo - workflow - write:packages - notifications - delete repo - write:discussions - project

To learn more about Scopes, visit the GitHub Scopes for OAUth Apps page.

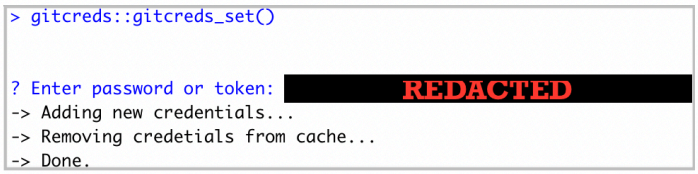
- Once Scopes are selected, click on the green `Generate token` button:



- Copy the generated PAT to your clipboard.
- Paste and Save this PAT in a text file in a secure folder that will NEVER be accessed by other users or the internet. You can create a `private` folder on your personal computer to store these files.
- Return to RStudio Console and run the following command:

`gitcreds::gitcreds_set()`.

You will be prompted to paste the PAT into the console:



Paste the PAT at the end of the line `Enter password or token:` and press enter.

- In the console, run:

```
usethis::git_sitrep()
```

This command should return your GitHub account information (see example below).

For more information on PATs, check out GitHub's PAT information page.

4.2.2 Git Learning Resource

- Happy Git and GitHub for the useR
- Atlassian Git Tutorial
- LinkedIn Learning
  - Git Essential Training: The Basics
  - Git from Scratch
  - Learning Git and GitHub
  - Git: Branches, Merges, and Remotes

Chapter 5

Project Management Tools

5.1 UCR VPN - GlobalProtect

UC Riverside provides free access to a Virtual Private Network (VPN), GlobalProtect, that allows you to access the CCB NAS Server or campus resources (ex: library services) when working remotely.

To install and use the GlobalProtect VPN, navigate to the UCR ServiceLink GlobalProtect Connection Installation Instructions.

5.2 CCB NAS Server

The Center for Conservation Biology has a NAS Server, which serves as the primary data storage location to access data and reference files. CCB staff should ensure that all work and research related files are saved on the NAS. Reference **Chapter** 7 for proper directory organization and file naming.

Network-attached storage (NAS) is dedicated file storage that enables multiple users to retrieve data from centralized disk capacity. The purpose of NAS is to enable users to collaborate and share data more effectively. It is especially useful to teams that need remote access. NAS connects to a wireless router, making it easy for remote workers to access files from any device with a network connection.

The CCB NAS server model is Synology DS920+. For more information on the unit, see **Chapter** 8.

5.2.1 Access the CCB NAS Server from Local Computer

In order to access the CCB NAS, there just a few steps:

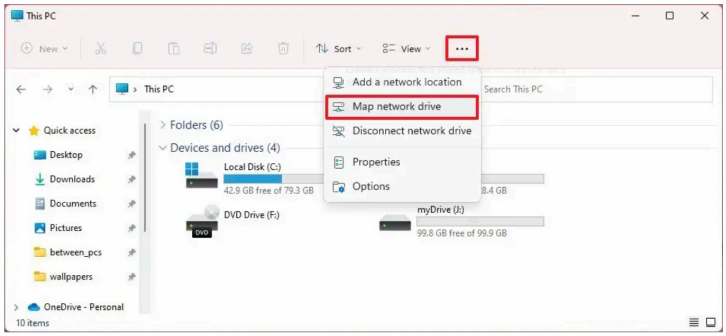
- **Step 01:** Request the creation of a NAS user account through the designated CCB staff member. You will need to provide your preferred **username** (typically similar to UCR ID) and a **passcode**. The passcode must be a strong code: a unique code with 12 - 15 characters consisting of a mix of letters, numbers, symbols, and does not contain personal information.

- **Step 02:** Once a user account has been created, log onto the UCR VPN (*see above section*).

- **Step 03:** Map the server onto your local computer:

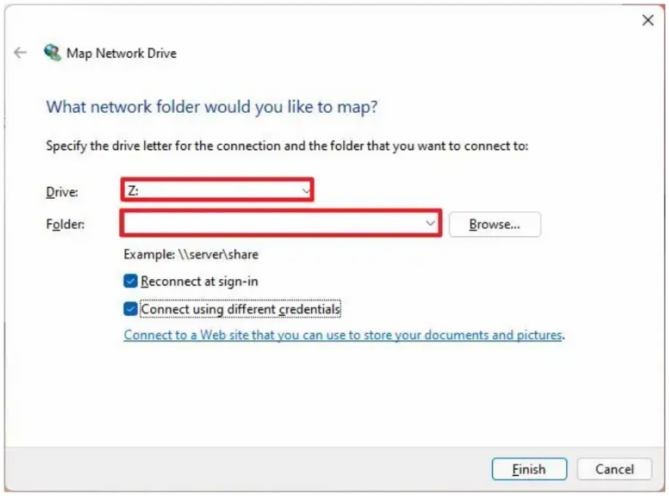
  – For PCs running Windows 11:

    1. Open File Explorer on Windows 11.
    2. Click on `This PC` from the left pane.
    3. Click the `See More` (three-dotted) button in the command bar and select the "Map network drive" option.



4. Use the `Drive` drop-down menu and choose a letter to assign the drive.

5. In the `Folder` field, enter the network path to the shared folder, `\\CCBNAS.dyn.ucr.edu\pdo`. (Or click the Browse button to browse the folder to map as a network drive, and click the OK button.)



6. Check the `Reconnect at sign-in` option to make the connection permanent.

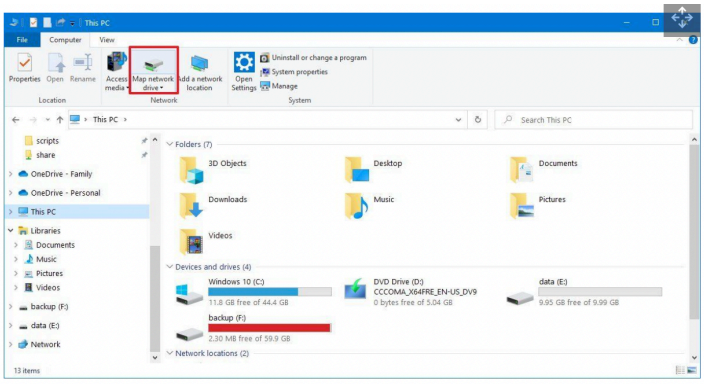
7. Check the Connect using different credentials option if the credentials are different from the account you are already using.
8. Click the Finish button.
9. Confirm the network account credentials (if applicable) to map the network drive to Windows 11.
10. Click the OK button.

Once you complete the steps, the network drive will become available in File Explorer.

If you run into issues, first read over this blog post for a common linkage issue.

- For PCs running Windows 10:

1. On your Windows 10 PC, open Windows Explorer.
2. Click on This Computer.
3. Click on Map network drive.



4. Select a drive letter from the drop-down menu.

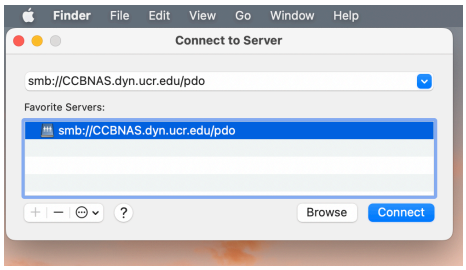
5. Enter `\\CCBNAS.dyn.ucr.edu\pdo` into the folder field.
6. Click Finish.
7. Enter your Synology NAS username and password in the Windows credentials pop-up prompt.

The mapped network drive will now appear within Windows Explorer as local storage, allowing you to quickly transfer files.

- For Macs:

1. Open the Finder app.
2. Click `Go` on top bar and select `Connect to Server`.
3. Type in the CCB NAS IP: `smb://CCBNAS.dyn.ucr.edu/pdo`
4. Enter `User Name` and `Passcode`.

There should be a connection set to the CCB NAS at the folder level.



5.2.2 Connecting the CCB NAS Synology to Git

The information below was taken from the Synology Knowledge Center.

Git is an open-source distributed version control system, which allows you to maintain software source code, documents, or any type of file on a computer with speed and efficiency.

5.2.2.1 To create a Git repository:

1. Sign in to DSM using an account with administrative privileges.
2. Go to Control Panel > Terminal & SNMP > Terminal then enable SSH service.
3. Go to Control Panel > Shared Folder and create a shared folder for Git repositories.
4. On your computer, enter the command below to access Synology NAS via SSH: `ssh [Synology NAS admin user name]@[Synology NAS IP address or hostname] -p [The port number of SSH]` For example, you can enter: `ssh myadminuser@192.168.1.2 -p 22`
5. Enter the command below to change the current directory to the shared folder you created in step 3: `cd /[Volume name]/[Shared folder name]/` For example, you can enter: `cd /volume1/mysharefolder/`
6. Enter the command below to create a folder on your computer for the Git repository: `mkdir [Folder name]`
7. Enter the command below to change the current location to the new folder: `cd [Folder name]`
8. Enter the command below to create a Git bare repository under the folder you created in step 6: `git init --bare`

Note:

- Please do not perform the above commands with root permission.

5.2.2.2 To Allow Users to Use Git on Synology NAS Server

1. Sign in to DSM using an account with administrators' privileges.
2. Go to **Control Panel > Terminal & SNMP > Terminal**, and enable **SSH service** for users to access Git repositories via SSH.
3. Go to **Control Panel > User & Group** and create a user. Grant **Read/Write** permission of the Git repository shared folder to the user.
4. Go to **Package Center > Installed** and open the **Git Server** package.
5. Allow the user to access repositories via git-shell.

Note:

- The permissions of default users (root, admin, and guest) cannot be edited.
- Apart from the user permission list for Git Server, all operations for Git Server should be performed via SSH instead of DSM desktop.
- Git users will be restricted to Git-related activities with a shell tool called git-shell. This login shell will be applied to Git users to ensure that the accounts are only used for Git operations. As a result, Git users may only use the SSH connection to push and pull Git repositories, and will not have full access to DSM.

5.3 Google Apps

The CCB team utilizes a number of Google applications for project management, data & file sharing, and communicating with colleagues.

Google Calendar: There is a lab Google calendar, *UCR CCB Palm Desert*. This calendar maintains lab mates work and leave schedules as well as any team meetings. Please make sure this calendar is shared with you once you have obtained your UCR NetId.

Google Drive: CCB utilizes *Shared Drive* folders as a file sharing application. You must request access to the shared drive folder from the lab P.I.

Google Sites:

The CCB's Desert Climate Research Initiative (DCRI) | Iniciativa de Resiliencia Climática del Desierto (IRCD) has a Google Sites that provides information on the Center's climate and community resilience work.

There is a CCB Google Site that is in draft form at the moment. This site may serve as an intranet or provide public facing information in the future.

5.4 Microsoft 360 Suite

UCR provides free access to Microsoft Office 365 Pro Plus for faculty, staff, and students! Please note that the available applications differ between PC and Mac OS. To learn more about downloading Microsoft applications, please visit the UC Riverside Microsoft 360 information page.

Microsoft 360 training resources can be found here.

5.5 Slack

Slack is a messaging app that connects teams via channels and direct messages (DMs). To make the most of this team communication tool, you will need to download the application to your desktop, have the application open throughout the workday, and adjust the notification settings to your preferences to receive timely alerts.

5.5.1 Download the Slack app

- For Windows: download Slack here.
- For Macs: download Slack here.
- Consider adding the app to your Android or iOS device(s) for mobile communication with the CCB team!

5.5.2 Join the CCB_UCR Slack Workspace

Follow this link to join the CCB Slack workspace. Once you have joined, you will be automatically be added to several workspace channels:

- `#general`: a channel for general announcements and team-wide conversations
- `#how_to_slack`: a channel for tips for how to best use Slack and its apps, as well as a space for the team to ask questions re: how to best utilize this communication tool. Check out the `pinned` messages and `bookmarked` websites.
- `#meetings`: a channel for announcing meetings and sharing resources for team meetings.
- `#random`: a channel to share random news, updates, ideas, jokes, etc.

5.5.3 Slack Resources

The Slack help center has useful *how to* guides and tutorials to facilitate working in Slack.

Slack Trainings

Check out Slack's **Top 5 tips for getting started in Slack** (5 min read) to learn the app's basic functions and communication workflows.

The Slack help center's "how to" guide has several sections:

- Quick Reference Guide (20 min read)
- Using Slack: recommended Sections to review
  - Format & style messages (5 min read)
  - Message features & tools (15 min read)
  - Audio & video (10 min read)
- Your Profile
  - Adjust your notifications: configure your notifications (5 min read)
  - Change your settings & preferences:
    - set your Slack status & availability (3 min read)
    - adjust your sidebar preferences (3 min read)
- Connect Tools
  - Connect tools from the Slack app directory: add apps to your Slack workspace (3 min read)
- Tutorials - a long list of tutorials to review as needed and as time allows

Slack Apps

Up to 10 Slack apps are available for free Slack accounts, and unlimited apps are available for Pro Slack accounts.

- A **Trello app** is incorporated in the CCB_UCR Slack workspace. You can create cards in Slack for the CCB Team Projects Trello workspace. To learn how, follow this Slack connect tool.
- The **Zoom app** can be added to Slack that allows your to start or join a Zoom Meeting or Zoom Phone call by using the **"/zoom"** slash command in any Slack channel or message.
- Add the **Google Drive app** to Slack to share resources from the CCB Shared Drive.
- There is an app for **Microsoft Teams** which allows a Teams video call from Slack.
  - Start a Teams video call right from Slack Launch a call in Slack with the /teams-calls slash command. Before joining, get a quick glance at who's already on the call and when the call kicked off.
  - Jump from Slack straight into a meeting — join Teams meetings directly from calendar reminders in Slack, using the Outlook or Google Calendar apps.
  - Customize call settings for your team. Set Microsoft Teams Calls as your default calling provider, so anyone can start a video call in Teams with a quick click of the phone button in Slack.

Other cool Slack workspaces to join:

- EcoDataScience - an environmental data science study group that started at UC Santa Barbara, but now has an international following!

- R4DS Online Learning Community

- Society for Open, Reliable, and Transparent Ecology and Evolutionary Biology (SORTEE)

5.6 Trello

Trello is a project management app tool that provides teams the opportunity to create task lists, reference & resource lists, and communicate via tagging! To connect with CCB's Trello Workspace, send a request to CCB's PI. The main board is **CCB Team Projects**.

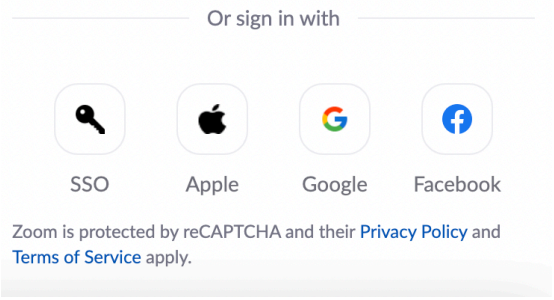
5.7 Zoom

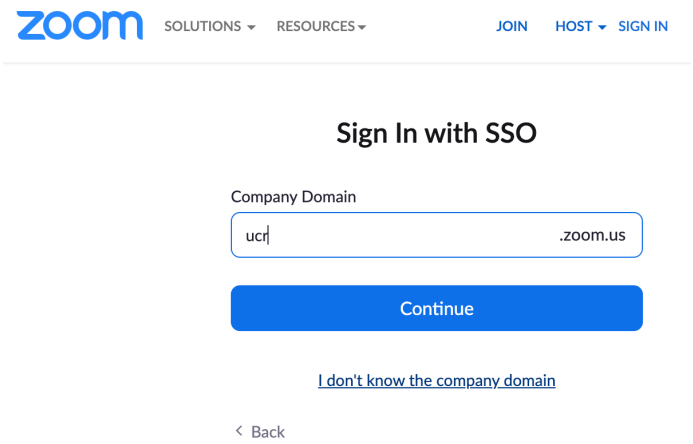
Zoom is a great tool to schedule virtual meetings, and the CCB hosts Zoom-based team meeting regularly. UC Riverside provides all faculty, staff, and students a free premium account. In order to utilize this service, download the application and sign in via Single Sign-ON (SS0). For the most current installation and log in information, visit UC Riverside's Information Technology Systems Zoom account page.

- **Step 1**: Download the Zoom application via UC Riverside's Zoom service page - just select the **Download** button!

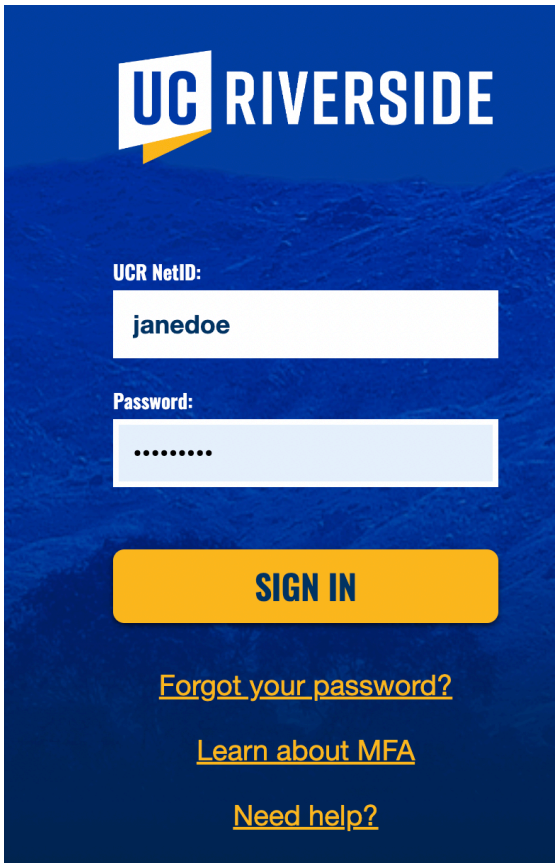
- **Step 2**: Once the application is downloading, Zoom will automatically open. You will be prompted to either *Join a Meeting* or *Sign In*. Select *Sign In* and set up your UCR Zoom account using your NetId information.

- **Step 3**: Click the *Sign In with SSO* button, then enter `ucr` in the *company domain* section.





- **Step 4**: You will then be prompted to sign in, this time using your UCR NetId. The program may open up an authentication window requesting your UCR NetId as shown below:



- **Step 5**: Personalize your Zoom account! You can add your name, pronouns, and an image to your profile, create a *personal Zoom room* to utilize as a default meeting space, and set settings like *mute all participants when they join a meeting* or turn off settings such as *start meetings with participant video on. Participants can change this during the meeting.*

Happy Zooming!

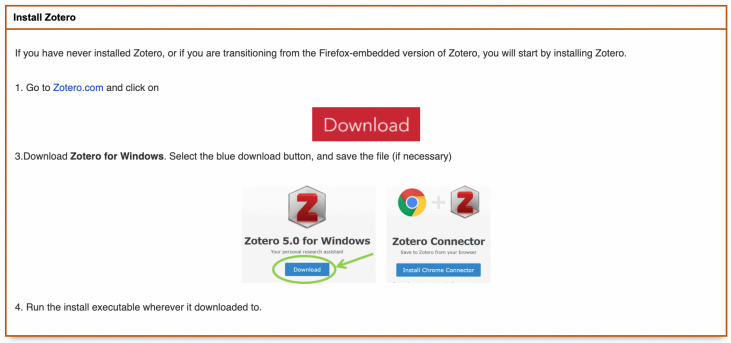
Zoom Resources:

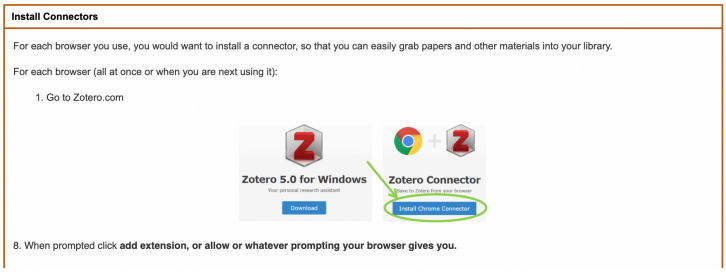
- Zoom Help Desk
- Zoom meetings trainings
- UC Riverside Zoom Services

5.8 Zotero

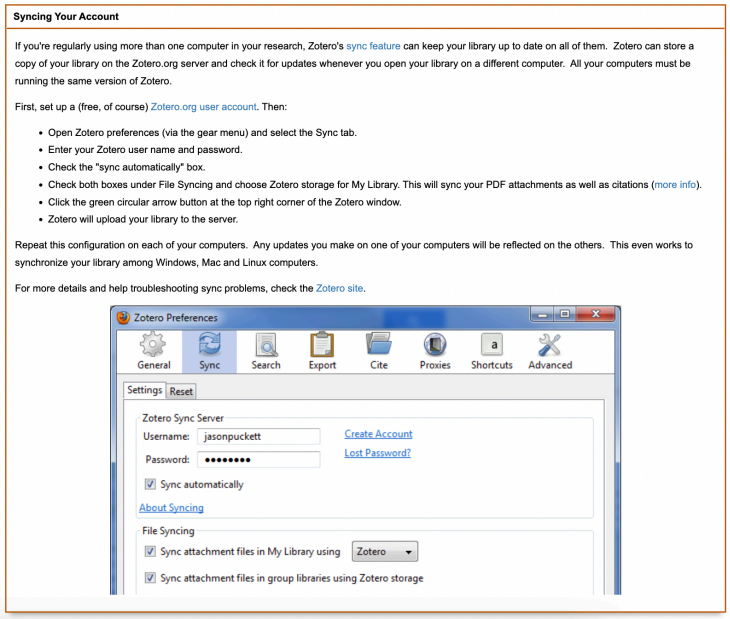
The Center for Conservation Biology shares research references through Zotero software. Zotero is a citation management tool and allows groups to collect and share references easily. The Zotero browser extension makes it easy to copy citations from webpages.

To install Zotero and the extension, please follow UC Riverside's installation instructions here. *Note*: These instructions are for a **Windows** machine, but it is a similar process for Macs.





The next step is to create a Zotero account and enable syncing to enable extension use and to be able to join the CCB Zotero Reference Group. Make sure to use your using the UCR email address when setting up the Zotero account.



To connect to the CCB Zotero Reference Group, navigate to this link. If you do not have access to the Group page, please send a request that you be added to the CCB PI.

For more information on how to use Zotero, check out these links:

- UC Riverside Zotero Tips & Tricks
- Zotero Quick Start Guide
- Zotero Groups
- RStudio Citations - add BibTex citation format in Zotero then quickly use .bib files in R to create references in markdown.

Chapter 6

ESRI GIS Platform

6.1 ESRI ArcGIS

UC Riverside provides free access to ESRI ArcGIS software and platforms for faculty, staff, and students. Learn more about these data analysis and visualization tools here.

ESRI Platform Software

- **ESRI ArcGIS Desktop**: a proprietary desktop GIS software suite that allows you to create maps, perform spatial analysis and manage data. Required software for installation: Microsoft .NET (aka a PC machine), Python 2.7+.
- **ESRI ArcGIS Pro**: ArcGIS Desktop is transitioning to ArcGIS Pro, an interactive spatial data visualization tool that allows users to create 2D maps, 3D scenes, and share projects on ArcGIS Online. Training resources for ArcGIS Pro are here.
- **ESRI ArcGIS Online**: This is cloud-based software that runs on any device with an internet connection. It allows users to easily publish maps on-line, access other UCR spatial products, and create spatial data dashboards and hubs. Check out UCR ArcGIS Online.

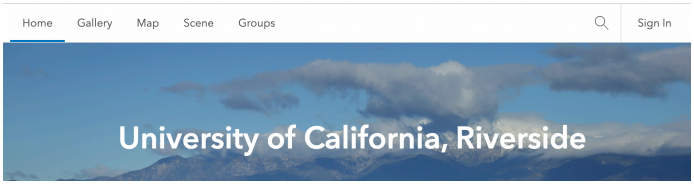
6.2 Install ESRI ArcGIS software

For additional installation information, navigate to UCR's ServiceLink Access ArcGIS Online and ArcGIS Pro help page.

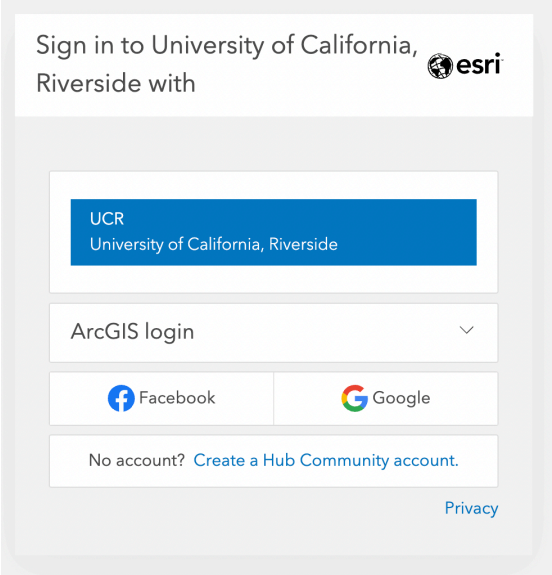
6.2.1 ESRI ArcGIS Online

Instructions for how to sign in to ArcGIS Online via UCR.

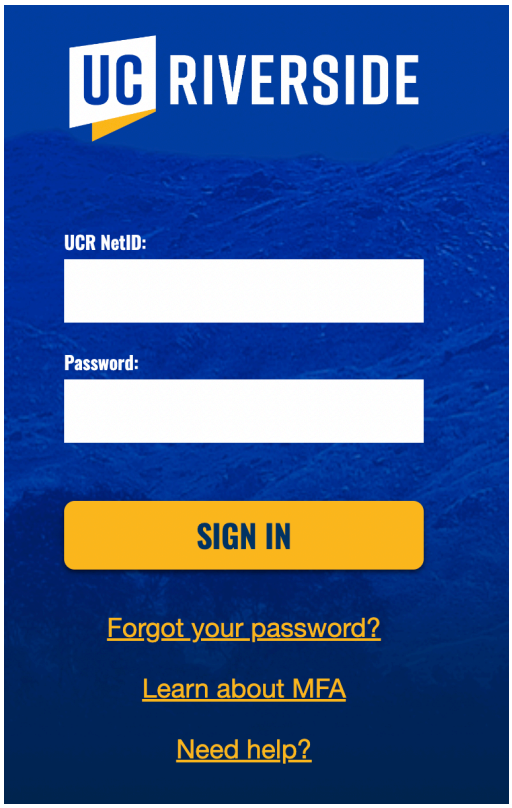
- **Step 1:** Navigate to the UCR's Academic ArcGIS Online site.
- **Step 2:** Click on the 'Sign In' link at the top right hand corner of the landing page.



- **Step 3:** Select the **UCR University of California, Riverside Enterprise login** (blue button) option.



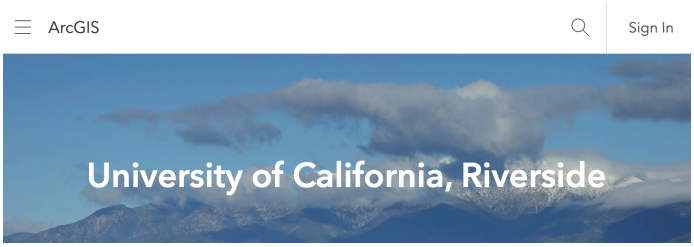
- **Step 4:** Sign in with UCR Net ID and Password



6.2.2 ESRI ArcGIS Pro

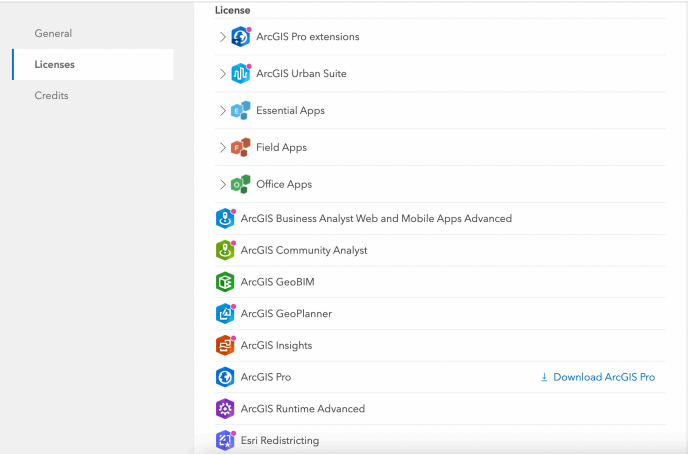
Instructions for how to download ArcGIS Pro from UCR's ArcGIS Online.

- **Step 1:** Navigate to the UCR's Academic Instance of ArcGIS Online.
- **Step 2:** Click on the `Sign In` link at the top right hand corner of the landing page.



- **Step 3:** Select the **UCR University of California, Riverside Enterprise login** (blue button) option under UCR.

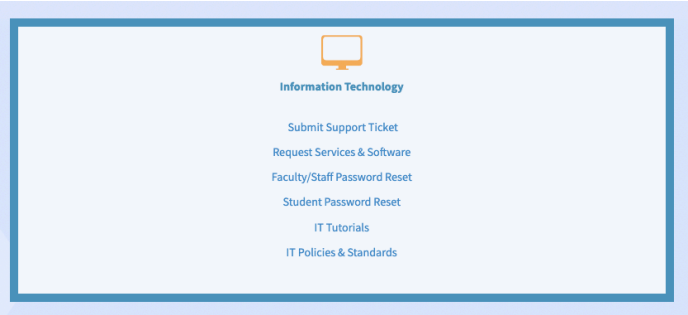
- **Step 4:** At the top right of the page, click your user name and click `My Settings`.
- **Step 5:** On the `My Settings` page, click the Licenses tab.
- **Step 6:** Scroll down, next to ArcGIS Pro, click `Download ArcGIS Pro`.



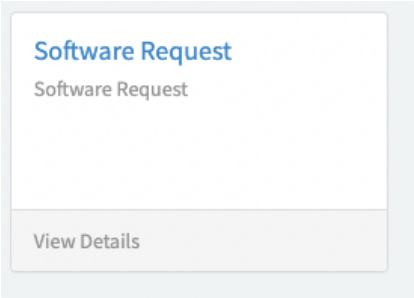
6.2.2.1 Archived UCR ESRI ArcGIS Desktop and Pro Installation Instructions

This was the installation process prior to 2020. If there are issues with the above instructions, this is an alternative method to access ESRI software.

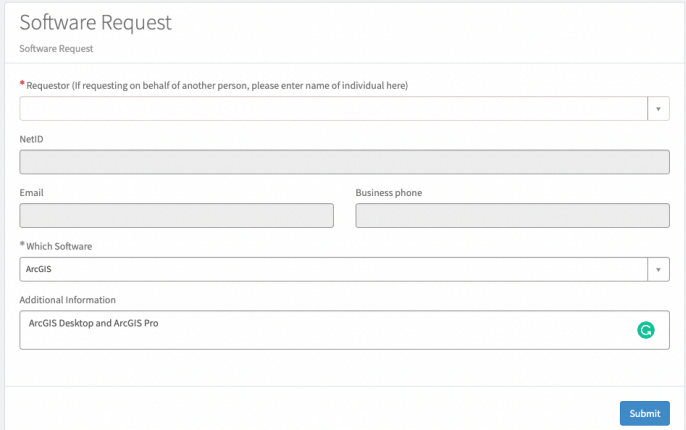
- **Step 1:** Navigate to the UCR Portal ServiceLink Software Catalog.
- **Step 2:** Under the Information Technology section, select **Request Services & Software** link.



- **Step 3:** Scroll down to the **Software Request** card and select link.



- **Step 4:** Complete the Software Request form using your UCR NetId and UCR email, then select **Submit**.



- **Step 5:** The UC IT team will send email notifications to provide further instructions.

6.2.3 ESRI Resources

Here are few resources that provide tutorials and workshops on how to use ESRI software:

- **ESRI Academy**: Search the ESRI Acadmey Course Catalog for MOOCs, Tutorials, and Web Courses.
- **UC ANR IGIS** - the UC Agriculture and Natural Resources Informatics & GIS Program host ESRI ArcGIS training webinars and virtual workshops throughout the year.
- **LinkedIn Learning** - Through UCR LinkedIn Learning, users may take ESRI ArcGIS courses and learning paths.

Chapter 7

Directory Organization & Data Project Development

7.1 File Naming

When creating file names, best practices dictate that you should:

- Create meaningful but brief names. Limit a file name to 25 characters, if possible;
- Use file names to classify types of files;
  - ex: `20230131_DTLA_RA.csv` file name classifies this file as Rapid Assessment vegetation monitoring data from the Desert Tortoise Linkage Area and was created on Jan. 31, 2023;
- Avoid using spaces, dots and special characters (ex: & or ? or !);
- Use capital letters to delimit words (aka UpperCamelCase), not spaces or underscores for **data files**.(ex: `20170217CVStormwaterVAP.csv`);

- Use dashes (-) or underscores (_) to separate elements in a file name for **documents** (ex: `20231212_CVMC_CRCA01_Proposal_LCS.doc`);
- Always use the ISO 8601 date format in file names: **YYYYMMDD**. For example, in the file name *20230131_DTLA_RA.csv* January 31, 2023 is represented as **20230131**;
- Preserve the 3-letter file extension for application-specific codes of file format (e.g. .doc, .xls, .mov, .tif);
- With sequential numbering (e.g., 1, 2, 3, etc.), use leading zeros to accommodate multi-digit versions. For example, use 01-10 for 1-10, 001-100 for 1-100, and so on;

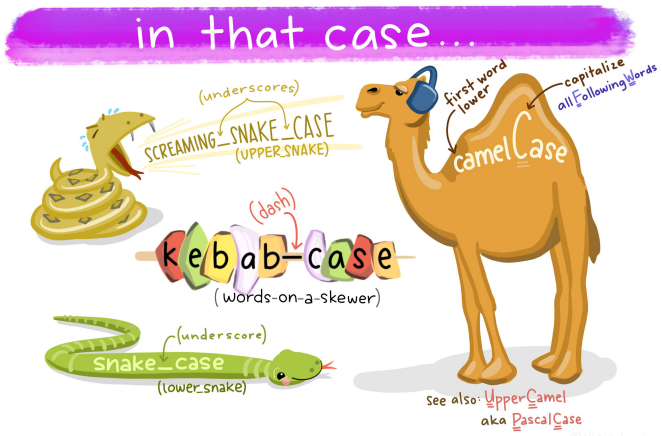


Figure 7.1: Types of font cases (L to R, clockwise): Screaming Snake Case, Camel Case, Upper Camel, Snake Case, Kebab Case. Art Credit: Dr. Allison Horst, UCSB

- Name all files to reflect their content or function. For example, use names such as `tortoise_count_table.csv`, `manuscript.Rmd`, or `vegetation_analysis.r`. Do **not** use general nomenclature with sequential numbers (e.g., `result1.csv`, `result2.csv`) or a location in a final manuscript (e.g., `fig_3_a.png`), since those numbers will almost certainly change as the project evolves;
- Include versioning of file names where appropriate, indicating the version with `rev` the date (ex: CVMC_CRCA01_Proposal_LCS_rev20230131.doc).
  - this is a bit different from the standardized **date first** file naming. It is recommended to **avoid** *starting* the filename with a version number .The CCB lab can set the versioning to `CVMC_CRCA01_Proposal_LCS_rev20230131.doc`, but for the final document, name it `20231212_CVMC_CRCA01_Proposal_LCS.doc`.
  - *Final copies of tabular data* should be stored as a copy without macros or formulas, in a non-proprietary format such as comma or tab separated values (.csv or .txt) (Smithsonian, 2021).
- Order the elements in a file name in the most appropriate way to retrieve the record. For example,
  - file name with the date first,
  - the project name or project location (acronyms are very helpful here) should be second,
  - then the type of data or file (either RA or Report or Brochure),
  - then version and/or author,

7.1.1 File Name Examples

- Vegetation assessment data collected on February 17, 2017 at the Coachella Valley Stormwater Channel: `20170217_CVStormwaterVAP.csv`,
- Coachella Valley Mountain Conservancy Climate Resistance & Community Access Grant Program (Proposal No. CRCA-01) proposal document authored by Lynn Sweet and finalized & submitted on December 31, 2022: `20221231_CVMC-CRCA01_Proposal_LCS.doc`
- Journal articles: `2020_Thorne_VegetationRefugiaInformClimate-adaptiveLandManagement.pdf`.

7.1.2 ESRI File Naming Guidelines & Unique Exceptions

When naming objects, folders, databases to be used by ESRI software, there are some unique considerations that contradict the above file naming guidelines!

- **Avoid long pathnames.** Try to "flatten" folder structures so that file pathnames under 128 characters. The absolute Windows pathname limit (MAXPATH) is 260 characters.
- **Start with a letter.** *Never* begin an ArcGIS object name (table, field, relationship class, mxd file, folder, etc) with a number or any other non-alpha character.
- **No reserved words, aka common coding function words.** Avoid names that may conflict with SQL or other language's reserved words. For example: `OBJECTID`, `VALUE`, `COUNT`, `NOT`, `OR`, `ON`, `IN`, `OVER`, `SELECT`. This is to avoid confusing both people and SQL or Python or R.
- **Be brief.** All field names should contain **10 characters or less**. This is a dBase limit, so it applies to .dbf files AND shapefiles. For that reason it's best practice in case you need to go "through" a dbf based format. Coverage and grid names have a limit of 13 characters, but 10 is safer. If you really want a longer name around or one with special characters, use the alias.
- **Raster dataset names MUST start with a letter** Raster names can be particularly problematic and it is best to keep the filename **under 14 characters** to support Esri grid format, and special characters in the path (space, &, -) may cause raster exports and tools to fail with cryptic error messages (like 999999).
- **Use code in scripting to check names.** The arcpy methods `ValidateFieldName` and `ValidateTableName` can be used at parameter validation time, or in your code, to prevent users of your tool from inserting invalid output names. The `CreateScratchName` method if provided the proper arguments, will generate valid dataset names.

7.1.2.1 Addtional Resources for Geodatabase File Management

- The United Nations Office for the Coordination of Humanitarian Affairs has an informative wiki page regarding how they organize and name spatial files.
  - The OCHA has an inspiring *dummy directory* structure here
- Earth Data Analysis Center created a helpful file naming and metadata creation presentation.
- UCSD GIS & Geospatial Resource list

7.2 Directory Organization

The amount of data and files research organizations create and store has grown significantly due to technological advances in field data collection tools and open-source software. As the amount of data collected and reports generated increases, it is necessary to establish a clear, logical directory organization framework.

7.2.1 Recommendations for Directory Organization

The following recommendations were pulled from Spreckelsen et al. (2020).

1. **Put each project in its own directory**

Create a new folder for each new project and name that folder after the project. Some researchers create a separate project for each manuscript they are working on, while others group all research on a common theme or data set into a single project. As a rule of thumb, divide work into projects based on the overlap in data and code files. If 2 research efforts share no data or code, they will probably be easiest to manage independently. If they share more than half of their data and code, they are probably best managed together, while if you are building tools that are used in several projects, the common code should probably be in a project of its own. Projects do often require their own organizational model, but below are general recommendations on how you can structure data, code, analysis outputs, and other files. The important concept is that it is useful to organize the project by the types of files and that consistency helps you effectively find and use things later.

2. **Put text documents associated with the project in a 'document' directory.**

This includes files for manuscripts, documentation for source code, and/or an electronic lab notebook recording your experiments. Subdirectories may be created for these different classes of files in large projects. Create subdirectories within the `document` folder to account for the various types of documents required for each project. For example, does this project include pre-proposals for funding? If so, create a subfolder `proposal`. If the project is grant funded, create a `report` folder to save the various reports required for the funding.

3. **Raw Data and Analysis Results**

- **Put raw data and metadata in a `data` directory**

The data directory might require subdirectories to organize raw data based on time, method of collection, or other metadata most relevant to your analysis. These files should include the raw .xls and .csv files, plus additional .txt & .md files.

- **Put files generated during cleanup and analysis in a `results` directory**

The `results` directory will usually require additional subdirectories for all but the simplest projects. *Generated files* are considered intermediate files, such as cleaned data, statistical tables, and final publication-ready figures or tables and should be separated clearly by file-naming conventions or placed into different subdirectorie.

4. **Put project source code in the `repository` directory.**

The `respository`, also contains all of the code written for the project. This includes programs written in interpreted languages such as R or Python; as well as shell scripts, snippets of SQL used to pull information from databases; and other code needed to regenerate the results. This directory may contain **two** conceptually distinct types of files that should be distinguished either by clear file names or by additional subdirectories.

- The first type is files or groups of files that perform the core analysis of the research, such as data cleaning or statistical analyses. These files can be thought of as the "scientific guts" of the project.
- The second type of file in `respository` is controller or driver scripts that contain all the analysis steps for the entire project from start to finish, with particular parameters and data input/output commands. A controller script for a simple project, for example, may read a raw data table, import and apply several cleanup and analysis functions from the other files in this directory, and create and save a numeric result. For a small project with 1 main output and all the code is in R, save R files in a `R` folder.

5. **Images and Visualizations**

Photos, screenshots, and plots or other types of data visualizations should be saved in a separate `images` project sub-directory.

6. **Put project scratch work and analysis in a `sandbox` directory.**

Create a `sandbox` directory within the appropriate project folder for experimental project work. A `sandbox` folder is useful in a the `src` or `r` folder as it can serve as location to save test code or tutorials to conduct analysis.

7. **Create an `archive` folder for out-of-date data or versions**

An `archive` folder serves as a repository for long-term data retention. Data and reference materials stored in the `archive` folder are no longer the most relevant versions, but important reference files that should be preserved but not utilized for analysis.

7.2.2 Folder Organization Examples:

1) Data Analysis Project

Simple Project

```
my-simple-project/
   InputData          <- Folder containing data that will be processed
   OutputData         <- Folder containing data that has been processed
   Figures            <- Folder containing with figures or tables summarizing the resul
   Code               <- Folder the scripts or programs to do the analysis
   LICENSE            <- File explaining the terms under which data/code is being made
   README.txt         <- File documenting the analysis, and (ideally) the purpose of ea
```

Advanced Projects

Many kinds of input data, documentation, and code.

```
my-advanced-project/
   repository/
   AUTHORS.md         <- File: List of people that contributed to the project (Markdown
   LICENSE            <- File: Plain text file explaining the usage terms/license of th
   README.md          <- File: Readme file (Markdown format)
   bin                <- Folder: Your compiled model code can be stored here (not track
   config             <- Folder: Configuration files, e.g., for doxygen or for your mod
```

```
   data               <- Folder: Data for this project
       external       <- Folder: Data from third party sources.
       interim        <- Folder: Intermediate data that has been transformed.
       processed      <- Folder: The final, canonical data sets for modeling.
       raw            <- Folder: The original, immutable data dump.
   docs               <- Folder: Documentation, e.g., doxygen or scientific papers (not tracked b
   notebooks          <- Folder: Ipython or R notebooks
   reports            <- Folder: Manuscript source, e.g., LaTeX, Markdown, etc., or any project r
       figures        <- Folder: Figures for the manuscript or reports
   src                <- Folder: Source code for this project
       R           <- Folder: scripts and programs to process data
       external       <- Folder: Any external source code, e.g., other projects, or external libr
       models         <- Folder: Source code for your own model
       tools          <- Folder: Any helper scripts go here
       visualization  <- Folder: Visualisation scripts, e.g., matplotlib, ggplot2 related
```

```
my-advanced-project_002/
   repository/
       images/
           image-01.jpg
           image-02.jpg
       templates/
           page.html
           post.html
       index.html
|    |   r/
     |      function.r
|    |      analysis.r
|    |      report.Rmd
       data/
     |      function.r
     |      function.r
       myproject.Rproj
   README.md
```

2) Vegetation Monitoring Project

```
vegetation/
DesertTortoiseLinkageArea/
   images/
       RelevePlots
           DTLA_plot001-01.jpg
           DTLA_plot001-02.jpg
       PlantId
           AlloniaIncarnata_BerdooCyn_2022-001.jpg
```

```
        AlloniaIncarnata_BerdooCyn_2022-002.jpg
repositories/
    DTLA
        r/
            function.r
            analysis.r
            report.Rmd
        data/
            processeddataplot001.csv
            processeddataplot002.csv
            rawdata/
                rawdataplot001.csv
                rawdataplot002.csv
        myproject.Rproj
        README.md
documents/
    DTLA
        _archive/
            geodatabase/
            report_drafts/
        reports/
            2021AnnualReport.doc
            2021AnnualReport.doc
            2021AnnualReport.doc
    reference/
        CNPS_RapidAssessmentProtocol/
            2017_CNPSRapidAssessmentProtocol.pdf
            2018_CNPSRapidAssessmentProtocol.pdf
            2020_CNPSRapidAssessmentProtocol.pdf
        Species/
            PlantIdentificationGuides
                2022_JOTRSpeciesGuide.pdf
            SpeciesInformation
                AlloniaIncarnata
                    AlloniaIncarnataSpeciesInfo.pdf
recycle/
```

7.2.3 Additional Resources

- Gentzkow M, Shapiro JM. Code and Data for the Social Sciences: A Practitioner's Guide; 2014. https://web.stanford.edu/~gentzkow/research/CodeAndData.pdf.
- Hampton S.E, et al. Skills and Knowledge for Data-Intensive Environmental Research. BioScience, Volume 67, Issue 6, June 2017, Pages 546–557.

- Noble WS. A Quick Guide to Organizing Computational Biology Projects. PLoS Comput Biol. 2009;5(7).

7.3 Data Management

Wilson et al. (2017) recommend following these data management practices:

- Save the raw data.
- Ensure that raw data are backed up in more than one location.
- Create the data you wish to see in the world.
- Create analysis-friendly data.
- Record all the steps used to process data.
- Anticipate the need to use multiple tables, and use a unique identifier for every record.
- Submit data to a reputable DOI-issuing repository so that others can access and cite it (ex: DataOne, KNB, and NCEI.

7.3.1 Tidy Data

Tidy data is one of the most important concepts for any data scientist. It provides predictable organization for data that makes coding, analysis, and collaboration easier.

Tidy data satisfies the following conditions:

- Each observation is in a row
- Each variable in a column
- Each value is in a single cell

Broman and Woo (2018) state the core principles for tidy data are:

- be consistent with data collection and entry,
- write dates like YYYY-MM-DD,
- don't leave any cells empty,
- put just one thing in a cell,
- organize the data as a single rectangle
  - with subjects as rows and variables as columns,
  - and with a single header row,
- create a data dictionary,
- don't include calculations in the raw data files,
- don't use font color or highlighting as data,
- choose good names for things,

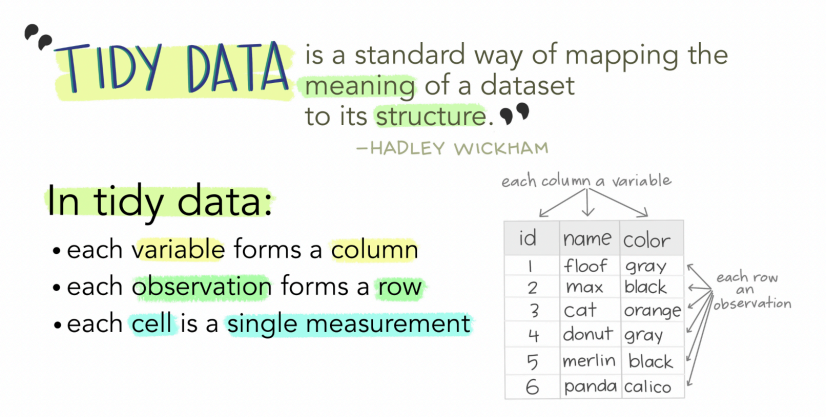


Figure 7.2: Tidy data allows for efficient data science, easy collaboration, and ensures reproducibility and reuse. Art Credit: Dr. Allison Horst, UCSB

- make backups,
- use data validation to avoid data entry errors,
- and save the data in **plain text files** (i.e. .csv, .txt).

The R package, `tidyverse`, is one of the best tools to tidy data! The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures.

Install the complete tidyverse with:

```
install.packages("tidyverse")
```

Then load the **tidyverse** package in a code chunk with the command:

```
library(tidyverse)
```

The packages contained within the **tidyverse** are:

- ggplot2 is a system for creating graphics for your data, based on The Grammar of Graphics. Plot your data with ggplot2!
- dplyr is a data manipulation package that provides a consistent set of verbs to manipilate data:
  - `mutate()` adds new variables that are functions of existing variables
  - `select()` picks variables based on their names.
  - `filter()` picks cases based on their values.
  - `summarise()` reduces multiple values down to a single summary.

  - `arrange()` changes the ordering of the rows.

- tidyr is the tidying data workhorse package within the **tidyverse**. Use **tidyr** to wrangle, pivot, and convert or replace missing data.
- readr is a fast way to read rectangular data from delimited files, such as comma-separated values (CSV) and tab-separated values (TSV).
- purrr provides a complete and consistent set of tools for working with functions and vectors.
- tibble is a version of `data.frame`, but they do less (i.e. they don't change variable names or types, and don't do partial matching) and complain more (e.g. when a variable does not exist).
- stringr provides a cohesive set of functions designed to make working with strings as easy as possible.
- forcats provides a suite of tools that solve common problems with factors, including changing the order of levels or the values.
  - `fct_reorder()`: Reordering a factor by another variable.
  - `fct_infreq()`: Reordering a factor by the frequency of values.
  - `fct_relevel()`: Changing the order of a factor by hand.
  - `fct_lump()`: Collapsing the least/most frequent values of a factor into "other".

7.3.2 Project Development and Collaboration

The foundation of developing and organizing a project is to design it to make it easy for new collaborators to join and for current coworkers to collaborate seemlessly. This involves making it easy for people to set up a local workspace so that they can contribute, help them find what to contribute, and make the organization process clear so that they know how to contribute.

Steps to consider when developing a research project:

1. **Create an overview of your project.**

Have a short file in the project's home directory that explains the purpose of the project. This file (generally README.txt or README.md) should contain the project's:

- title,
- a brief description,
- up-to-date contact and/or author information,
- and an example or 2 of how to run various cleaning or analysis tasks,
- describes what is needed for the project and use or contribute to it, i.e., dependencies that need to be installed, tests that can be run to ensure that software has been installed correctly, and guidelines or checklists that the project adheres to.

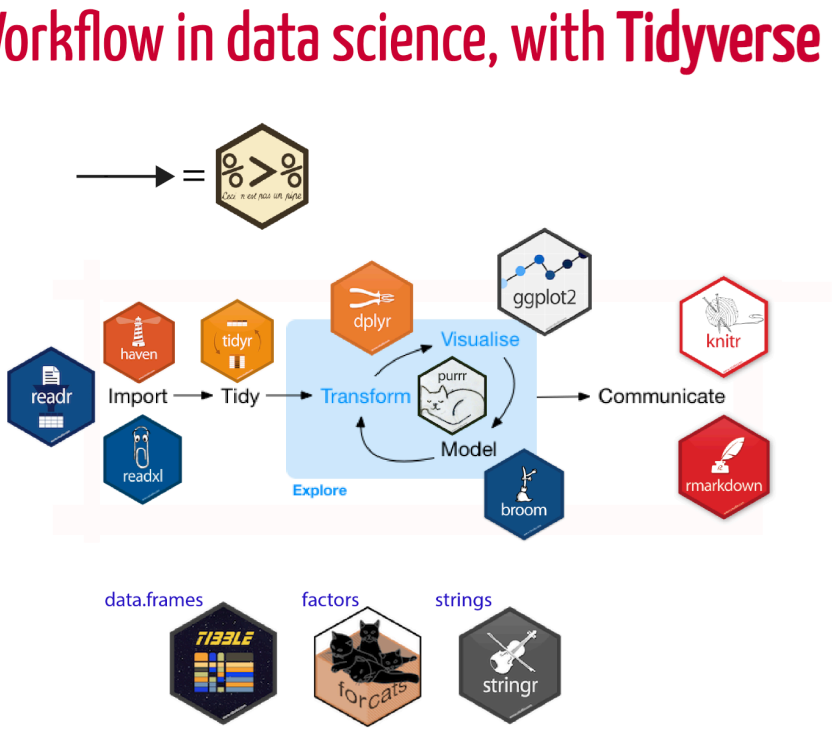


Figure 7.3: Data Science workflow utilizing Tidyverse packages. Art Credit: Dr. Olivier Gimenez

It is often the first thing users and collaborators on your project will look at, so make it explicit how you want people to engage with the project.

2. **Create a shared "to-do" list.**

This can be a plain text file called something like notes.txt or todo.txt, or utilize GitHub project management tools to create a new issue for each to-do item. Describe the items clearly so that the tasks are clear and easily understood.

3. **Decide on communication strategies.**

Make explicit decisions about (and publish where appropriate) how members of the project will communicate with each other and with externals users/collaborators. This includes the location and technology for email lists, chat channels, voice/video conferencing, documentation, and meeting notes, as well as which of these channels will be public or private.

4. **Make the license explicit.**

Create a LICENSE file in the project's home directory that clearly states what license(s) apply to the project's software, data, and manuscripts. The one most frequently recommended is the **Creative Common** licenses for data and text.

5. **Ensure the project is citable.**

Include a `CITATION` file in the project's home directory that describes how to cite this project as a whole and where to find (and how to cite) any data sets, code, and figures. Below is an example `CITATION` file for the Ecodata Retriever (https://github.com/weecology/retriever):

```
Please cite this work as: Morris, B.D. and E.P. White. 2013. "The
EcoData Retriever: improving access to existing ecological data".
PLoS ONE 8:e65848.
```

Guidance above taken from Wilson et al. (2017).

7.3.3 FAIR Data Management Principles

In 2016, the **FAIR Guiding Principles for scientific data management and stewardship** were published in *Scientific Data*. The authors intended to provide guidelines to improve the **F**indability, **A**ccessibility, **I**nteroperability, and **R**euse of digital assets (Wilkinson et al., 2016).

In this article, the authors outline the **FAIR** Guiding Principles as:

1. To be **F**indable:
    - F1. (meta)data are assigned a globally unique and persistent identifier
    - F2. data are described with rich metadata (defined by R1 below)
    - F3. metadata clearly and explicitly include the identifier of the data it describes
    - F4. (meta)data are registered or indexed in a searchable resource
2. To be **A**ccessible:
    - A1. (meta)data are retrievable by their identifier using a standardized communications protocol
        - A1.1 the protocol is open, free, and universally implementable
        - A1.2 the protocol allows for an authentication and authorization procedure, where necessary
    - A2. metadata are accessible, even when the data are no longer available
3. To be **I**nteroperable:
    - I1. (meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.
    - I2. (meta)data use vocabularies that follow FAIR principles
    - I3. (meta)data include qualified references to other (meta)data
4. To be **R**eusable:
    - R1. meta(data) are richly described with a plurality of accurate and relevant attributes
        - R1.1. (meta)data are released with a clear and accessible data usage license
        - R1.2. (meta)data are associated with detailed provenance
        - R1.3. (meta)data meet domain-relevant community standards

7.3.4 Additional Resources

For more information on tidy data and data management:

- Julia Lowndes & Allison Horst's Tidy Data blog
- Hadley Wickham Tidy Data
- Grolemund & Wickham R for Data Science
- Lowndes, J.S., et al. Our Path to Better Science in Less Time Using Open Data Science Tools.

Chapter 8

Equipment

Inventory of software and equipment used and references

- ARC MAP DESKTOP: ArcMap ArcGIS Desktop, version 10.9 (Esri, Redlands, California)
- LASER RANGEFINDER: Hilti PD-42 laser rangefinder (Hilti Corporation, Schaan, Liechtenstein)
- TABLETS: Samsung Galaxy Tab Active 2 tablet (Samsung Group, Seoul, South Korea)
- GPS: Trimble Juno 3
- SERVER:
  - NAS model – Synology DS920+
  - Storage – HUS728T8TALE6L4 Ultrastar DC HC320 8TB (x2, one for backup)
  - OS & Monitoring software – Synology DSM (DiskStation Manager) 7.1-42661

Chapter 9

Concur Travel

UCR provides reimbursements for travel, lodging, and meals associated with travel for work. All travel must be discussed and approved by the PI (Lynn Sweet)

The new Concur Travel system has a modern, mobile-friendly user interface with built-in business rules to check for completeness prior to submitting an Expense Report. Please remember that, even though the travel application is changing, the UC travel policy is not and the Expense Report in the new Concur Travel system will still require a business purpose and itemized receipts for travel expenses. And, similar to the old iTravel system, the new Concur Travel system will allow only the Traveler to approve a Declaration of Missing Evidence (DME) and *only* the Traveler can submit an Expense Report.

The new Concur Travel system uses a two-step process:

1 - Submitting a Trip Request prior to travelling

2 - Submitting an Expense Report after your trip is complete (Please note that BMPN Travel Arrangers can assist with training for this step)

Note You can reach out to the BMPN Travel Arrangers (Sarah Acrey, Jeanette Westbrook, or Elisha Hankins) for assistance in completing your first Expense Report in Concur Travel. Please also note that at this time travel arrangers are not made aware when a trip has been requested or expense report submitted. While campus works to create a report, I ask that you please send a quick email to BMPNPurchasing@ucr.edu to notify us that you have a travel pending review to prevent delays in processing.

9.1 Submitting a Trip Request - *Before the trip*

First, you will need to log into the Concur Travel system.

-Log into your UCR Portal - https://portal.ucr.edu/.

-Navigate to Authorized Apps, then select the icon for Concur Travel and Expense.

-Log in with your verified UCR email address, the one associated with your UCR account, and sign in with university credentials

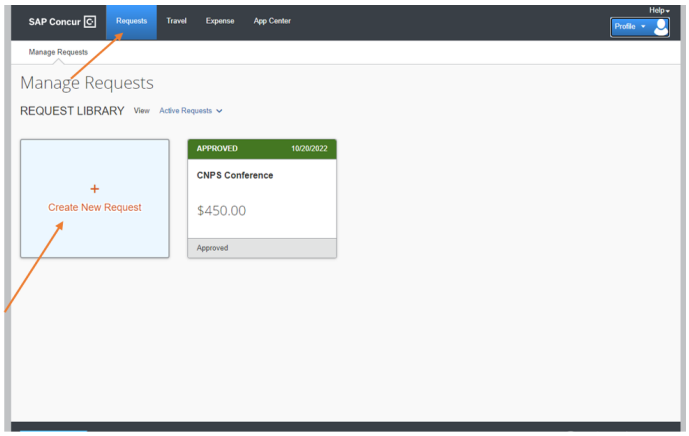
9.1.1 Make a Trip Request Profile

-Select *Requests* on the upper right of the home page

-Select *Create New Requests* in the center of the page

-Fill out the appropriate information for the request

-Click *Create New Request*



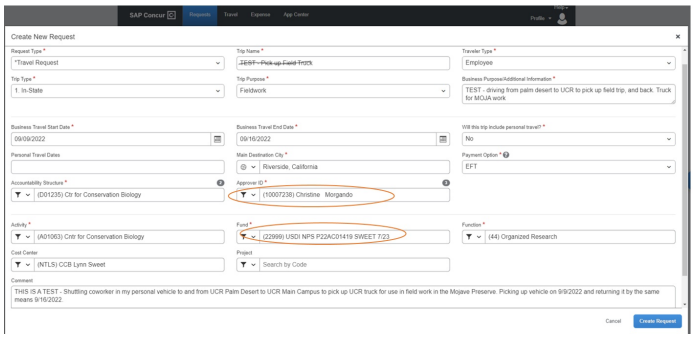
IMPORtANT NOTE

-Trip Name must follow this format: Request ID - Travel start date - Travel end date (ex 34kf – 082822 – 090122). If you cannot find the Request ID, you can add it to the name after you have made the request under the title

-'Fund' is the FAU number. For the appropriate FAU see the Trello tab with current codes

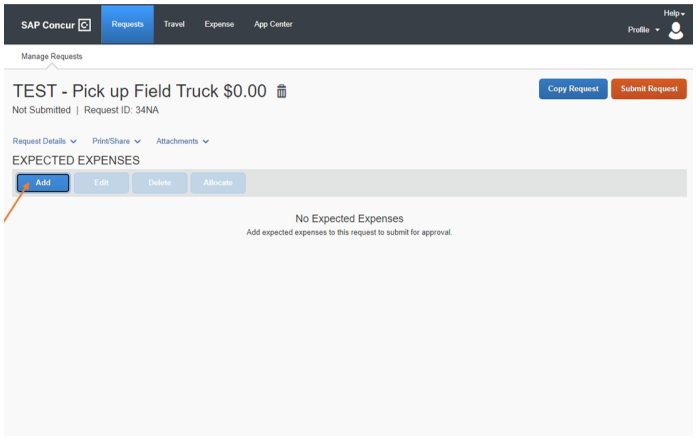
-Cost center might change based on projects (*NTCB* is for projects where Cam was the PI when they were created and *NTLS* is for projects where Lynn is the PI),

-Approver ID is 'Christine Morgando'.

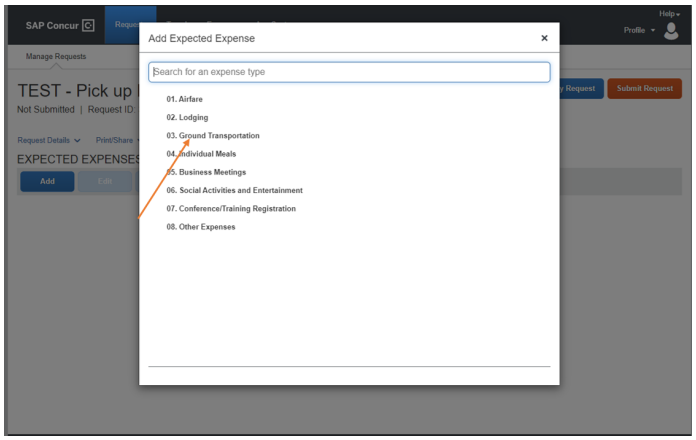


9.1.2 Adding Expenses to the Trip Request

-Select the *Add* button under Expected Expenses

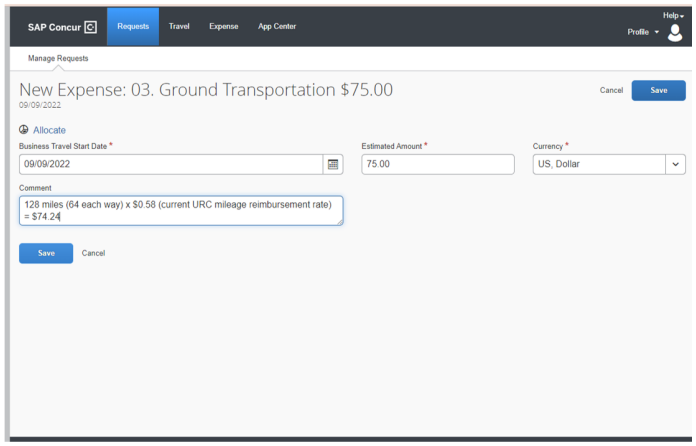


-Select *Ground Transportation* (or any other *expected* expenses such as food, lodging, registrations, etc.)

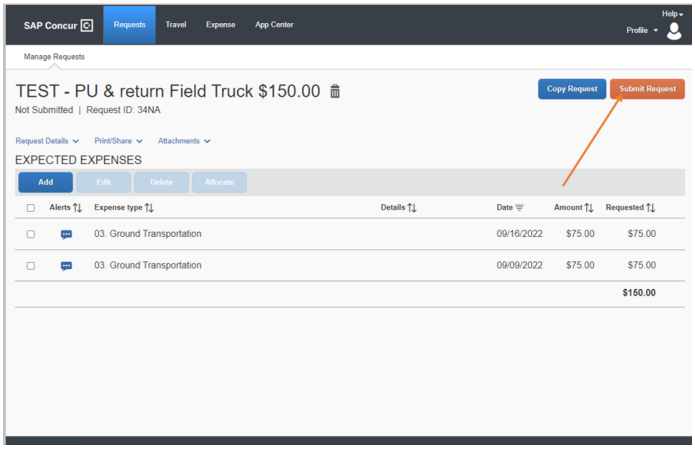


-Create a *New Expense* for the trip

-Fill out the page with estimated amounts you expect for that category and Save *NOTE: MILEAGE REIBURSEMENT RATE HAS CHANGED, CURRENT RATES CAN BE FOUND [HERE] (https://accounting.ucr.edu/travel-entertainment/mileage-reimbursement-rates)*



-Continue adding expected expenses until you've covered your bases, and then submit request.



-Email Debbie Brown (debbie.brown@ucr.edu) with the subject line being the Request ID to notify her that you put in a request so she can approve it

9.2 Submitting an Expense Report - *After the trip*

-From the request page, open the Approved request

-Select Create Expense Report

THIS IS AS FAR AS I HAVE GOTTEN AS OF 9/9/2022, UPDATES COMING SOON - MEL

Chapter 10

Code of Conduct

The Center of Conservation Biology's Code of Conduct adapted from (Perry, 2018)

10.1 We are committed to working as a team.

All aspects of our professional contributions to the project are discussed and agreed upon together, and all tasks - although they might be led by individual team members - are developed through collaborative practice. Devotion to supporting the team, working as a team player, providing constructive critique to your team members, and respecting the interests of the team as a successful working group (without compromising their safety or security, as described below), are paramount.

10.2 We are committed to prioritizing and championing the people and communities that host us.

Our work is driven by local needs, and decision-making is grounded in evidence and robust data gathered in local contexts. We are critically aware of the existing evidence. We attend events and participate in activities that are organised by our host communities. We respect, care for and create long-lasting friendships with our hosts. We aim to abide by local expectations around dress and custom, and if working in communities where the primary language is not our own, we are committed to learning the language. We maintain links with our hosts after the project ends and we support their future professional endeavors.

10.3 We are committed to the working hours, professional expectations and responsibilities defined by the overall project directors.

We typically work as part of a larger project team guided by wider goals than ours alone. We are aware of their responsibilities, we have read the necessary guidance documents, we have understood and signed the necessary insurance and risk assessment documentation, and in all cases, we respect and abide by the instructions given by the directors. This includes zero tolerance in relation to behavior that compromises the well being, equality, security or dignity of other human beings, as described below.

10.4 We are representatives and extensions of the University of California, Riverside and its staff, and of the professional bodies to which we and our project leaders are subscribed.

We recognize our duty of care to, and our responsibility for professionalism in, not only the communities where we work and reside, but the university and surrounding organisations to which we and our project leaders are accountable. Our behaviors reflect on these institutions and we acknowledge that our direct supervisor is (and therefore we too are) bound by the ethical and professional codes of both UC Riverside and other institutional affiliations.

Considering these obligations, you agree with the following:

- I will come to my direct supervisor the moment that I experience problems, challenges or trouble of any kind. I will keep her informed of any issues that I feel may manifest themselves in relation to myself, my teammates or affiliates while in the field. If I feel I need support beyond my direct supervisor, I will turn to the 2nd identified lead for their advice. I have already disclosed to my direct supervisor any potential matters of concern (which may include matters relating to health, psychological and physical wellbeing, security, equality, confidence, interpersonal relations, previous travel or fieldwork experiences, etc.) so that they are aware of them and can mitigate them prior to departing for - and during - fieldwork. If I have not yet disclosed such matters, I agree to do so as soon as possible. I have shared this information in confidence, with an expectation of complete privacy unless urgent medical, safety, security, or other legal intervention is required.

10.5 We recognize that fieldwork can be intense, emotional and tiring.

We understand that things can go wrong, that we may need to compromise, and that in exceptional circumstances, we made need to shorten or modify your work on site to help manage these circumstances. In such cases, we will have a series of conversations about how to deal with difficulties, led by your direct supervisor and/or other lead. If the difficulties are not resolved within 7 days of identification, we will consult with the university for their guidance. If it is agreed with the University that the difficulties are unresolvable in the field, we will help you to organize your safe return home.

10.6 We have the right to a safe, secure and non-threatening working and living environment.

We do not tolerate any form of discriminatory, abusive, aggressive, harassing, threatening, sexually- or physically-intimidating, or related problematic behaviors that compromise the wellbeing, equality, security or dignity of other human beings (whether those humans are our peers, colleagues, supervisors, collaborators, local community members or any persons at all). Our supervisors are trained in supporting those who have experienced or are experiencing harassment. They are obliged to investigate and respond to observed, implied or directly reported harassment. Considering this zero-tolerance policy, you agree to the following:

- I will not engage in behavior that compromises the wellbeing, equality, security or dignity of other human beings. I recognize that if I am implicated in such behavior I will be required to leave the project at my own expense and may be subject to criminal investigation.

If I witness others being subjected to such behavior, I will report it immediately to my direct supervisor. If I feel I cannot speak to my direct supervisor, I will report it to the 2nd identified lead. If I feel I cannot report it to either my direct supervisor or the 2nd lead, then I will contact the University of California, Riverside Compliance office.

- If I myself feel unsafe or uncomfortable, I will report it immediately to my direct supervisor. My supervisor will support me and will implement actions to keep me safe while working to stop the behavior. If I feel I cannot speak to my direct supervisor, I will report it to the 2nd identified lead. If I feel I cannot report it to either my direct supervisor or the 2nd

lead, then I will contact the University of California, Riverside Compliance office.

- My commitment to creating and maintaining safety and security for all extends to my online (web and social media) and mobile phone interactions, and I recognize that the process for reporting and acting on threatening online/mobile phone behaviors is the same as above.

10.7 Important University of California, Riverside Links

- UCR Values
- Ethics & Compliance Program
- Equal Opportunity and Affirmative Action
- Discrimination, Harassment, and Retaliation Complaint and Resolution
- Standards of Conduct
- Environmental Heath & Safety
- Alcohol and Drug Policy
- Personal Relationship Policy

Bibliography

Broman, K. W. and Woo, K. H. (2018). *Data organization in spreadsheets*. Number e3183v2.

Perry, S. (2018). Six fieldwork expectations: Code of conduct for teams on field projects.

Smithsonian, L. . A. (2021). Best practices for filenaming, organizing, and working with data.

Spreckelsen, F., Rüchardt, B., Lebert, J., Luther, S., Parlitz, U., and Schlemmer, A. (2020). Guidelines for a standardized filesystem layout for scientific data. *Data*, 5(22):43.

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., Gonzalez-Beltran, A., Gray, A. J. G., Groth, P., Goble, C., Grethe, J. S., Heringa, J., 't Hoen, P. A. C., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S. J., Martone, M. E., Mons, A., Packer, A. L., Persson, B., Rocca-Serra, P., Roos, M., van Schaik, R., Sansone, S.-A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M. A., Thompson, M., van der Lei, J., van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J., and Mons, B. (2016). The fair guiding principles for scientific data management and stewardship. *Scientific Data*, 3(11):160018.

Wilson, G., Bryan, J., Cranston, K., Kitzes, J., Nederbragt, L., and Teal, T. K. (2017). Good enough practices in scientific computing. *PLOS Computational Biology*, 13(6):e1005510.

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2022). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.29.

Yıldırım, S. (2020). What is git and why is it so important?