Using R and RStudio on the Cluster

Introduction: This workshop will discuss how the various R related tools and RStudio work together on the cluster and introduce a series of best practices for managing these environments.

Course Goals:

- Using R from the command line.
- Refresh on creating ‘R’ conda environments.
- Where are libraries installed?
- Using RStudio via OnDemand.
    - Using a Conda environment within RStudio
- Introduction aspects of using R in parallel on the cluster.

This is not a workshop on learning the R language, but on how to use R on the cluster.

Notes:

- The workshop modules work best in a sequential manner as a story introducing concepts and providing examples, but sections can be used separately to focus on a particular concept.
- You will need to *modify* usernames, project names, and folder locations, to apply to yourself.

---

10. Using R/RStudio on the Cluster: Summary: Summarize the concepts covered across the workshop.

Where are R Packages Installed on the Cluster?

Goal: Understand where R installs packages and where libraries are located, as well as inspecting general R system configuration.

Terminology Package vs Library

R-Bloggers: Packages v. Libraries in R: **Packages** are collections of R functions, data, and compiled code in a well-defined format. The directory where packages are stored is called the **library**:

- A package is a directory of files which extend R, either a source package (the master files of a package), or a tarball containing the files of a source package, or an installed package, the result of running `R CMD INSTALL` on a source package. On some platforms there are also binary packages, a zip file or tarball containing the files of an installed package which can be unpacked rather than installing from sources.
- A package is **not** a library. The latter is used in two senses in R documentation. The first is a directory into which packages are installed, e.g. `/usr/lib/R/library`: in that sense it is sometimes referred to as a library directory orlibrary tree (since the library is a directory which contains packages as directories, which themselves contain directories). …

Load an R Environment via the Module System

```
[salexan5@mblog1 ~]$ module load gcc/13.2.0 r/4.4.0
[salexan5@mblog1 ~]$ R --version
R version 4.4.0 (2024-04-24) -- "Puppy Cup"
...

# Run a simple function.
[salexan5@mblog1 ~]$ R -e "print(version[['version.string']])"
R version 4.4.0 (2024-04-24) -- "Puppy Cup"
...
> print(version[['version.string']])
[1] "R version 4.4.0 (2024-04-24)"

# Run an R script.
[salexan5@mblog1 ~]$ Rscript r_test.R
[1] "R version 4.4.0 (2024-04-24)"
```

Inspect R Environment Configuration

```
[salexan5@mblog1 ~]$ R
R version 4.4.0 (2024-04-24) -- "Puppy Cup"
...
# List ALL environment variables:
> Sys.getenv()

# Get single environment variable.
> Sys.getenv("R_HOME")
[1] "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R"

# Get a list of environment variables.
> Sys.getenv(c("R_PLATFORM", "R_HOME", "R_LIBS_USER"))
   R_PLATFORM  "x86_64-pc-linux-gnu"
   R_HOME      "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R"
   R_LIBS_USER "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4"
```

Notice that User Libraries are stored under your home, in: `R/x86_64-pc-linux-gnu-library/4.4`

In general, under `~/R/<R_PLATFORM>/versionX.Y/`

Get R Related Environment variables

Quick and dirty way to list all `R_*` environment variables.

```
[salexan5@mblog2 ~]$ R -e "Sys.getenv()" | grep R_
LMOD_FAMILY_COMPILER_VERSION
...
R_HOME                  /apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R
R_INCLUDE_DIR           /apps/u/spack/gcc/13.2.0/r/4.4.0-
pvzi4gp/rlib/R/include
R_LIBS_SITE             /apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/site-
library
R_LIBS_USER             /cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4
...
R_PLATFORM              x86_64-pc-linux-gnu
...
R_SHARE_DIR             /apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/share
...
R_UNZIPCMD              /usr/bin/unzip
R_ZIPCMD                /usr/bin/zip
thunderer_FAMILY_COMPILER_VERSION
```

Where are Packages Installed

```
> help(".libPaths")
.Library                 package:base                  R Documentation
Search Paths for Packages
Description:
     '.libPaths' gets/sets the library trees within which packages are
     looked for.
[salexan5@mblog2 ~]$ R -e ".libPaths()"
...
> .libPaths()
[1] "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-library/4.4"
[2] "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"

# Compare against:
R_LIBS_USER: /cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-library/4.4
R_LIBS_SITE: /apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/site-library
```

```
/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-library/4.4
```

This is where, for this R platform and version, your packages will be installed.

```
/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library
```

This is where the **base packages** for this R platform/version are installed.

You will not have permissions to install into this location.

```
[salexan5@mblog2 ~]$ ls /apps/u/spack/gcc/13.2.0/r/4.4.0-
pvzi4gp/rlib/R/library
base  compiler  datasets  graphics  grDevices  grid  methods  parallel
splines  stats  stats4  tcltk  tools  translations  utils
```

How to Install Packages (from within R)

```
# Within R:
> help(install.packages)
install.packages             package:utils             R Documentation
Install Packages from Repositories or Local Files
Description:
     Download and install packages from CRAN-like repositories or from
     local files.
Usage:
     install.packages(pkgs, lib, repos = getOption("repos"),
                      contriburl = contrib.url(repos, type),
                      method, available = NULL, destdir = NULL,
                      dependencies = NA, type = getOption("pkgType"),
                      configure.args = getOption("configure.args"),
                      configure.vars = getOption("configure.vars"),
                      clean = FALSE, Ncpus = getOption("Ncpus", 1L),
                      verbose = getOption("verbose"),
                      libs_only = FALSE, INSTALL_opts, quiet = FALSE,
                      keep_outputs = FALSE, ...)
...
```

How to Install Packages (from outside of R)

```
[salexan5@mblog2 ~]$ R CMD INSTALL --help
Usage: R CMD INSTALL [options] pkgs

Install the add-on packages specified by pkgs.  The elements of pkgs can
be relative or absolute paths to directories with the package
sources, or to gzipped package 'tar' archives.  The library tree
to install to can be specified via '--library'.  By default, packages are
installed in the library tree rooted at the first directory in
.libPaths() for an R session run in the current environment.

Options:
  -h, --help            print short help message and exit
  -v, --version         print INSTALL version info and exit
  -c, --clean           remove files created during installation
...
```

Note: “*The elements of pkgs can be relative or absolute paths to directories with the package sources, or to gzipped package 'tar' archives.*”

i.e. You have the package already downloaded.

Install tidyr Package

```
# Within R:
> install.packages("tidyr")
Installing package into ‘/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4’
(as ‘lib’ is unspecified)
--- Please select a CRAN mirror for use in this session ---
...
also installing the dependencies ‘utf8’, ‘generics’, ‘pillar’, ‘R6’,
‘stringi’, ‘fansi’, ‘pkgconfig’, ‘withr’,
‘cli’, ‘dplyr’, ‘glue’, ‘lifecycle’, ‘magrittr’, ‘purrr’, ‘rlang’, ‘stringr’,
‘tibble’, ‘tidyselect’, ‘vctrs’, ‘cpp11’
...
* DONE (tidyr)
```

Lets check:

```
[salexan5@mblog2 ~]$ ls /home/salexan5/R/x86_64-pc-linux-gnu-library/4.4
cli  cpp11  dplyr  fansi  generics  glue  lifecycle  magrittr  pillar
pkgconfig
purrr  R6  rlang  stringi  stringr  tibble  tidyr  tidyselect  utf8  vctrs
withr
```

What’s Installed?

```
# Within R:
> write.table(installed.packages()[,c(1,2,3:4)])
"Package" "LibPath" "Version" "Priority"
"cli" "cli" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-library/4.4"
"3.6.3" NA
"cpp11" "cpp11" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "0.4.7" NA
...
"tcltk" "tcltk" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
"tools" "tools" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
"utils" "utils" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
```

Note what is identified as 'base'.

Full List:

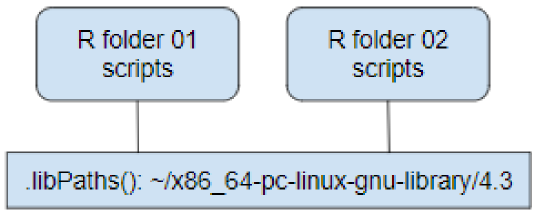
```
> write.table(installed.packages()[,c(1,2,3:4)])
"Package" "LibPath" "Version" "Priority"
"cli" "cli" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-library/4.4"
"3.6.3" NA
"cpp11" "cpp11" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "0.4.7" NA
```

```
"dplyr" "dplyr" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.1.4" NA
"fansi" "fansi" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.0.6" NA
"generics" "generics" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "0.1.3" NA
"glue" "glue" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.7.0" NA
"lifecycle" "lifecycle" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.0.4" NA
"magrittr" "magrittr" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "2.0.3" NA
"pillar" "pillar" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.9.0" NA
"pkgconfig" "pkgconfig" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "2.0.3" NA
"purrr" "purrr" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.0.2" NA
"R6" "R6" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-library/4.4"
"2.5.1" NA
"rlang" "rlang" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.1.4" NA
"stringi" "stringi" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.8.4" NA
"stringr" "stringr" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.5.1" NA
"tibble" "tibble" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "3.2.1" NA
"tidyr" "tidyr" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.3.1" NA
"tidyselect" "tidyselect" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-
gnu-library/4.4" "1.2.1" NA
"utf8" "utf8" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.2.4" NA
"vctrs" "vctrs" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "0.6.5" NA
"withr" "withr" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "3.0.0" NA
"base" "base" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
"compiler" "compiler" "/apps/u/spack/gcc/13.2.0/r/4.4.0-
pvzi4gp/rlib/R/library" "4.4.0" "base"
"datasets" "datasets" "/apps/u/spack/gcc/13.2.0/r/4.4.0-
pvzi4gp/rlib/R/library" "4.4.0" "base"
"graphics" "graphics" "/apps/u/spack/gcc/13.2.0/r/4.4.0-
pvzi4gp/rlib/R/library" "4.4.0" "base"
"grDevices" "grDevices" "/apps/u/spack/gcc/13.2.0/r/4.4.0-
pvzi4gp/rlib/R/library" "4.4.0" "base"
"grid" "grid" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
"methods" "methods" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
"parallel" "parallel" "/apps/u/spack/gcc/13.2.0/r/4.4.0-
pvzi4gp/rlib/R/library" "4.4.0" "base"
"splines" "splines" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
```

```
"stats" "stats" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
"stats4" "stats4" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
"tcltk" "tcltk" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
"tools" "tools" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
"utils" "utils" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
```

Remember: R Versions and Library Locations

Use Case: If projects/scripts are using say: `module load /4.3.x`:

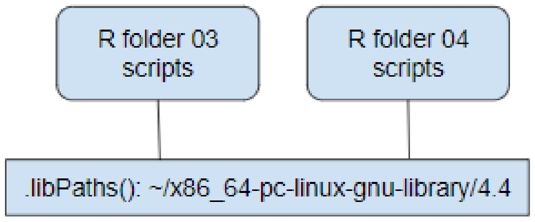


The scripts in folder 01/folder 02 will both use/share the R packages under `~/x86_64-pc-linux-gnu-library/4.3`

If you update an R package due to a need for a script in `folder 01`, then this new R package will also be used by the scripts in `folder 02`.

Is this intended? Does it course an issue for scripts in `folder 02`? You need to be aware and manage.

Similarly: If projects/scripts are using: `module load r/4.4.x`:



The scripts in folder 03/folder 04 will both use/share the R packages under `~/x86_64-pc-linux-gnu-library/4.4`

If you update an R package due to modifying a script in `folder 03`, then this new R package will also be used by the scripts in `folder 04`.

They will not be using any `r/4.3.x` related packages - that's a *different* library location.

R Conda Environments and Installed Packages

Goal: Understand R environments build with Conda.

Note: There is a known vulnerability with R versions less than 4.4.0.

This page uses R version 4.3.3. This page is purely for example since as of the date of creating this page there were issues using an r-base/4.4.1.

I am hoping in a few weeks/months the 4.4.x base will be stable and I'll update the examples.

Basic R Conda Environment

General Process:

```
[salexan5@mblog2 ~]$ cd /project/arcc/software/conda-envs/
[salexan5@mblog2 conda-envs]$ module load miniconda3/24.3.0
[salexan5@mblog2 conda-envs]$ conda search r-base
Loading channels: done
```

```
# Name                       Version          Build  Channel
...
r-base                         4.3.3      hf0d99cb_1  conda-forge
...
r-base                         4.4.1      h1dca405_0  conda-forge
[salexan5@mblog2 conda-envs]$ conda create -p r_4.3.3_env r-base=4.3.3
[salexan5@mblog2 conda-envs]$ conda activate
/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 conda-envs]$ R --version
R version 4.3.3 (2024-02-29) -- "Angel Food Cake"
...
```

Where are Packages Installed?

Lets look at the environment variables:

```
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 conda-envs]$ R -e 'Sys.getenv(c("R_PLATFORM", "R_HOME",
"R_LIBS_USER", "R_LIBS", "R_LIBS_SITE"))'
R version 4.3.3 (2024-02-29) -- "Angel Food Cake"
...
> Sys.getenv(c("R_PLATFORM", "R_HOME", "R_LIBS_USER"))
  R_PLATFORM  "x86_64-conda-linux-gnu"
  R_HOME      "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R"
  R_LIBS_USER "/cluster/medbow/home/salexan5/R/x86_64-conda-linux-gnu-
library/4.3"
  R_LIBS      ""
  R_LIBS_SITE "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/site-library"
```

Note: The platform string value is different: `x86_64-conda-linux-gnu`

Try Installing `vctrs` Package

```
# Within R:
> install.packages("vctrs")
...
also installing the dependencies ‘cli’, ‘glue’, ‘lifecycle’, ‘rlang’
...
* DONE (vctrs)
```

Where was this installed?

Check `.libPaths()`:

```
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 ~]$ R
R version 4.3.3 (2024-02-29) -- "Angel Food Cake"
...
> .libPaths()
[1] "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library"
```

Since I ***own*** this Conda environment, I ***have permission*** to install under: `/project/arcc/software/conda-envs/r_4.3.3_env/lib/R`.

Lets check:

```
[salexan5@mblog2 ~]$ ls /project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library/
base  cli  compiler  datasets  glue  graphics  grDevices  grid  lifecycle
methods
parallel  rlang  splines  stats  stats4  tcltk  tools  translations  utils
vctrs
```

But What about the `R_LIBS_USER` Environment Variable?

Remember our environment variables:

```
R_LIBS_USER "/cluster/medbow/home/salexan5/R/x86_64-conda-linux-gnu-
library/4.3"
R_HOME      "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R"
[salexan5@mblog2 ~]$ ls /home/salexan5/R/x86_64-conda-linux-gnu-library/4.3
ls: cannot access '/home/salexan5/R/x86_64-conda-linux-gnu-library/4.3': No
such file or directory
[salexan5@mblog2 ~]$ ls /home/salexan5/R/
x86_64-pc-linux-gnu-library
```

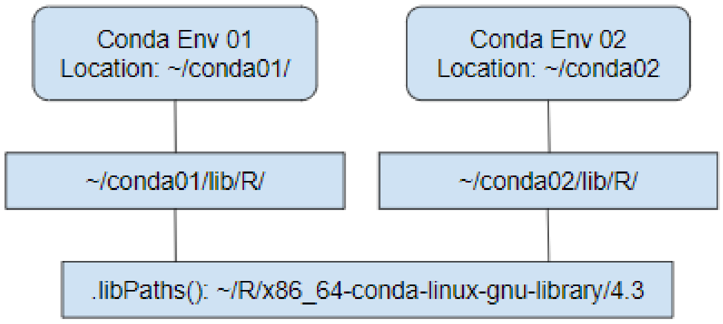
Since this folder is missing, it is not being picked up and used by the `.libPaths()` command.

Can I Create this Folder?

If you *manually* create this folder then `.libPaths()` does pick it up.

```
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 R]$ mkdir -p x86_64-conda-linux-gnu-library/4.3
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 R]$ R
R version 4.3.3 (2024-02-29) -- "Angel Food Cake"
...
> .libPaths()
[1] "/cluster/medbow/home/salexan5/R/x86_64-conda-linux-gnu-library/4.3"
[2] "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library"
```

Using the `install.packages()` command within R will ***now*** look/install packages into this folder that could be ***shared*** across multiple Conda environments using this version of R.



Should I Create this Folder?

Question: Should you manually create the `~/R/x86_64-conda-linux-gnu-library/X.Y/` folder?

First, we would suggest that unless you are confident in ***self-managing*** your Conda/R environments then do not.

If this is not in your `.libPaths()`, then it will force all R package installs to be contained under the Conda environment.

If this path ***IS*** in your `.libPaths()`, across multiple conda environments, then all the environments will install into and look/share this folder.

You can run into ***dependency/version issues*** if you want to use version X of a package in one Conda environment, but version Y in another. *You can only have one version in this folder*.

We have observed that some times this folder have been ***automatically*** created.

Best thing is to ***always*** check and set the `.libPaths()` to you necessary needs.

Try Installing `stringi` Package

```
# Within R:
> install.packages("stringi")
...
configure: error: in `/tmp/RtmpVm7ias/R.INSTALL2ad60710733cd2/stringi':
configure: error: C++ compiler cannot create executables
See `config.log' for more details
ERROR: configuration failed for package ‘stringi’
* removing ‘/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library/stringi’

The downloaded source packages are in
        ‘/tmp/Rtmpio5YFH/downloaded_packages’
Updating HTML index of packages in '.Library'
Making 'packages.html' ... done
Warning message:
In install.packages("stringi") :
  installation of package ‘stringi’ had non-zero exit status
```

Conda environments are not the same as the System/compute nodes - libraries and behavior can be different.

Conda Install `stringi` Package: Search

```
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 conda-envs]$ conda search stringi
Loading channels: done
No match found for: stringi. Search: *stringi*
# Name                       Version           Build  Channel
r-stringi                      0.4_1        r3.1.3_0  pkgs/r
...
r-stringi                      1.8.4   r43hbd1cc82_0  conda-forge

(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 conda-envs]$ conda search r-stringi
Loading channels: done
# Name                       Version           Build  Channel
r-stringi                      0.4_1        r3.1.3_0  pkgs/r
...
r-stringi                      1.8.4   r43hbd1cc82_0  conda-forge
```

Remember: Conda r packages use the naming convention: `r_<package-name>`

Conda Install `stringi` Package

```
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 conda-envs]$ conda install r-stringi
Channels:
 - conda-forge
 - defaults
Platform: linux-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##
  environment location: /cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env
  added / updated specs:
    - r-stringi

The following NEW packages will be INSTALLED:
  r-stringi          conda-forge/linux-64::r-stringi-1.8.4-r43hbd1cc82_0
...
Executing transaction: done
```

Where was this installed?

```
[salexan5@mblog2 ~]$ ls /project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library/
base  cli  compiler  datasets  glue  graphics  grDevices  grid  lifecycle
methods  parallel
rlang  splines  stats  stats4  stringi  tcltk  tools  translations  utils
vctrs
```

Anaconda: R Essentials

Although we do not recommend installing Anaconda (unless you are comfortable with how it works and modifies your home environment), it does provide an R Essentials bundle which "*includes approximately 80 of the most popular scientific packages for the R programming language.*"

```
conda install -c r r-essentials
```

R Packages and System Modules

Goal: Installing some R packages requires understanding what libraries are available on the System.

Start a new Session and Start R

```
[salexan5@mblog2 ~]$ module load gcc/13.2.0 r/4.4.0
[salexan5@mblog2 ~]$ R
R version 4.4.0 (2024-04-24) -- "Puppy Cup"
...
```

Try Installing R `XML` Package

```
> install.packages("XML")
...
checking for xml2-config... /apps/u/spack/gcc/13.2.0/libxml2/2.10.3-
5toq4pi/bin/xml2-config
...
Located parser file -I/apps/u/spack/gcc/13.2.0/libxml2/2.10.3-
5toq4pi/include/libxml2 -I/apps/u/spack/gcc/13.2.0/libiconv/1.17-
3dj22ny/include/parser.h
Checking for 1.8:  -I/apps/u/spack/gcc/13.2.0/libxml2/2.10.3-
5toq4pi/include/libxml2 -I/apps/u/spack/gcc/13.2.0/libiconv/1.17-
3dj22ny/include
Using libxml2.*
checking for gzopen in -lz... yes
checking for xmlParseFile in -lxml2... yes
You are trying to use a version 2.* edition of libxml but an incompatible
library. The header files and library seem to be
mismatched. If you have specified LIBXML_INCDIR, make certain to also specify
an appropriate LIBXML_LIBDIR if the
libxml2 library is not in the default directories.
ERROR: configuration failed for package 'XML'
* removing '/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4/XML'
...
```

What does the error message mean?

How to Resolve XML Package Issue

We can see from the CRAN package XML description that from the System Requirements we require: `libxml2 (>= 2.6.3)`

We can see that the `libxml2` library is available and is being picked up.

```
[salexan5@mblog2 ~]$ ml
Currently Loaded Modules:
...
 15) libxml2/2.10.3                56) nghttp2/1.57.0
...
```

Install location: `/apps/u/spack/gcc/13.2.0/libxml2/2.10.3-5toq4pi`

Sometimes the cause of the error isn’t obvious.

Try Googling to look for suggestions: Can not install XML package

```
# Withi R:
> Sys.setenv(XML_CONFIG="/apps/u/spack/gcc/13.2.0/libxml2/2.10.3-
5toq4pi/bin/xml2-config")
> install.packages('XML', configure.args=c('--with-
libxml2=/apps/u/spack/gcc/13.2.0/libxml2/2.10.3-5toq4pi/bin/xml2-config
LIBXML_LIBDIR=-L/apps/u/spack/gcc/13.2.0/libxml2/2.10.3-5toq4pi/lib/
LIBXML_INCDIR=/apps/u/spack/gcc/13.2.0/libxml2/2.10.3-
5toq4pi/include/libxml2/'))
Installing package into ‘/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4’
...
* DONE (XML)
```

Install the R `sf` Package

Reading the CRAN sf definition page, take note of the system requirements:

System Requirements: GDAL (>= 2.0.1), GEOS (>= 3.4.0), PROJ (>= 4.8.0), sqlite3

What do we already have loaded?

```
[salexan5@mblog2 ~]$ ml
Currently Loaded Modules:
...
 35) sqlite/3.43.2                 76) scrnsaverproto/1.2.2
...
```

We also need to manually load:

```
[salexan5@mblog2 ~]$ module load gdal/3.7.3 geos/3.12.0 proj/9.2.1
```

Install the R `sf` Package: Fails

```
> install.packages("sf")
...
also installing the dependencies ‘proxy’, ‘MASS’, ‘e1071’, ‘class’,
‘KernSmooth’, ‘wk’, ‘classInt’, ‘DBI’, ‘Rcpp’, ‘s2’, ‘units’
...
-----------------------------------------------------------------------------
---
  Configuration failed because libudunits2.so was not found. Try installing:
    * deb: libudunits2-dev (Debian, Ubuntu, ...)
    * rpm: udunits2-devel (Fedora, EPEL, ...)
    * brew: udunits (OSX)
  If udunits2 is already installed in a non-standard location, use:
    --configure-args='--with-udunits2-lib=/usr/local/lib'
  if the library was not found, and/or:
    --configure-args='--with-udunits2-include=/usr/include/udunits2'
  if the header was not found, replacing paths with appropriate values.
  You can alternatively set UDUNITS2_INCLUDE and UDUNITS2_LIBS manually.
-----------------------------------------------------------------------------
---
ERROR: configuration failed for package ‘units’
...
```

Do we have a `udunits` module?

```
[powersw@mblog2 ~]$ module spider udunits
-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
---------------------
  udunits: udunits/2.2.28
-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
---------------------
    You will need to load all module(s) on any one of the lines below before
the "udunits/2.2.28" module is available to load.
      arcc/1.0  gcc/13.2.0
    Help:
      Automated units conversion
[powersw@mblog2 ~]$ module load udunits/2.2.28

# Within R
> install.packages("sf")
...
* DONE (sf)

> library(sf)
Linking to GEOS 3.12.0, GDAL 3.7.3, PROJ 9.2.1; sf_use_s2() is TRUE
```

Recommendation: When ever you want to use this R package, additionally load:

```
[salexan5@mblog2 ~]$ gcc/13.2.0 r/4.4.0
[salexan5@mblog2 ~]$ module load gdal/3.7.3 geos/3.12.0 proj/9.2.1
udunits/2.2.28
```

Creating a Shared Library of R Packages

Goal: Demonstrate how to use an R library to create a shared set of R packages.

Use Case

Consider the following use cases:

- You're collaborating on some research and want users of the project to use the same R environment and set/version of a collection of R packages.
- You're leading a workshop and want all attendees to learn using the same environment.

In both cases we can setup an R library within a shared location, such as a ***project*** folder, which all users can access, and thus use the same set of packages.

General Process

The general process for this is:

1. Create a folder in a shared location.
2. Load and start R.
3. Update the library paths to point to this location.
4. Install R packages.

Every time this is to be used:

1. Load and start R.
2. Update the library paths to point to this location.

Example

```
# Create R Library folder:
[salexan5@mblog2 ~]$ cd /project/arcc/software/
[salexan5@mblog2 software]$ mkdir -p r_library/r_workshop
[salexan5@mblog2 r_library]$ cd r_workshop/
[salexan5@mblog2 r_workshop]$ pwd
/project/arcc/software/r_library/r_workshop

# Load and Start R
[salexan5@mblog2 ~]$ module load gcc/13.2.0 r/4.4.0
[salexan5@mblog2 ~]$ R

# Check current library paths:
> .libPaths()
[1] "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-library/4.4"
[2] "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
```

Create Current Available R Packages

```
> write.table(installed.packages()[,c(1,2,3:4)])
"Package" "LibPath" "Version" "Priority"
"class" "class" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "7.3-22" "recommended"
...
"XML" "XML" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-library/4.4"
"3.99-0.17" NA
"base" "base" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
...
"utils" "utils" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
```

Note: We can current see packages installed:

1. Under our home folder.
2. Base packages installed as part of this R version.

Update the Library Path

```
> .libPaths(c('/project/arcc/software/r_library/r_workshop',
'/apps/u/spack/gcc/12.2.0/r/4.4.0-7i7afpk/rlib/R/library'))
> write.table(installed.packages()[,c(1,2,3:4)])
"Package" "LibPath" "Version" "Priority"
"base" "base" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
...
"utils" "utils" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
```

Note: We can currently only see the base packages since we haven't yet installed anything into `/project/arcc/software/r_library/r_workshop`

Install a package into the New Library Location

```
# Within R:
> install.packages("Matrix")
...
also installing the dependency 'lattice'
...
* DONE (Matrix)

> write.table(installed.packages()[,c(1,2,3:4)])
"Package" "LibPath" "Version" "Priority"
"lattice" "lattice"
"/cluster/medbow/project/arcc/software/r_library/r_workshop" "0.22-6"
"recommended"
"Matrix" "Matrix"
"/cluster/medbow/project/arcc/software/r_library/r_workshop" "1.7-0"
"recommended"
"base" "base" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
...
"utils" "utils" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"

# Lets check:
[salexan5@mblog2 r_library]$ ls r_workshop/
lattice  Matrix
```

Test and Use

Error:

```
[salexan5@mblog1 r_library]$ cat r_library_test_fail.R
library(Matrix)

[salexan5@mblog1 r_library]$ Rscript r_library_test_fail.R
Error in library(Matrix) : there is no package called ‘Matrix’
Execution halted
# The Matrix package is NOT available under our default library location.
```

Success:

Update the `.libPaths()` every time you wish to use this R library.

```
[salexan5@mblog1 r_library]$ cat r_library_test.R
.libPaths(c('/project/arcc/software/r_library/r_workshop',
'/apps/u/spack/gcc/12.2.0/r/4.4.0-7i7afpk/rlib/R/library'))
library(Matrix)

[salexan5@mblog1 r_library]$ Rscript r_library_test.R
[salexan5@mblog1 r_library]$
```

Warning: Remember

Within this example, by default this is a **SHARED *project*** location.

Anyone who has access to this ***project*** and thus this shared library can update it.

If an individual performs an `install.packages()` it will effect everyone.

The project users will need to collaborate and come up with some form of data plan and/or process for the management of this shared library.

Using R and RStudio within OnDemand

Goal: Detail the process of using R and RStudio via the OnDemand service.

- What is RStudio
- Start R and RStudio

What is RStudio

RStudio Desktop: Used by millions of people weekly, the RStudio integrated development environment (IDE) is a set of tools built to help you be more productive with R and Python.

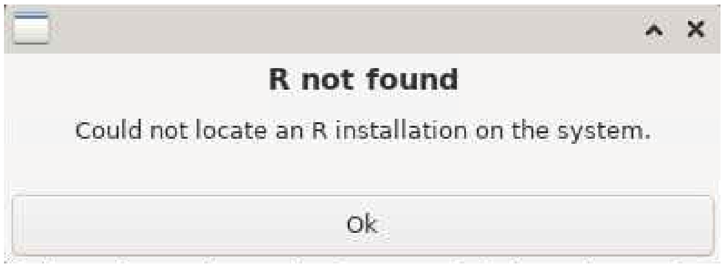
Since this is a desktop GUI application, you will need to run this from ARCC’s OnDemand service, via an interactive desktop.

Start R and RStudio

Process:

1. Access OnDemand and start an interactive desktop.
2. Open up a terminal.
3. Load R.
4. Load RStudio
5. Call `rstudio` from the command line.

RStudio requires a version of R to be available, so a version of R must be loaded before trying to start RStudio.



R not found

Could not locate an R installation on the system.

Using an R Conda Environment with RStudio

Goal: Detail how to use an R Conda Environment within RStudio.

- Can I use conda install rstudio?

Can We?

As detailed in this post: Should/can you run R Studio in Conda?

Yes, you can launch RStudio to use R that was installed inside a conda environment.

But Posit does not officially support this use case, so it might require some tinkering to get it working on your machine.

The process that will be detailed works as of time of writing this workshop.

We will monitor this process - but if this doesn’t work then please contact ARCC.

General Process

We will use the Conda Environment that we created in: Basic R Conda Environment

Process:

1. Access OnDemand.
2. Start an Interactive Desktop.
3. Open a terminal.
4. From the command-line set:
5. `# Using the path to your conda environment.`
6. `# In this case: /project/arcc/software/conda-envs/r_4.3.3_env`
7.
```
export PATH=$PATH:/project/arcc/software/conda-envs/r_4.3.3_env/bin/
export RSTUDIO_WHICH_R=/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/bin/R
```
8. Load `rstudio`.
9. Start `rstudio` from the command-line.

What did we do?

Using the path to the conda environment in which we setup R, we extended the `PATH` environment to this conda environments bin folder, and then set the `RSTUDIO_WHICH_R` environment variable to the R executable within this environment.

In General:

```
1. export PATH=$PATH:<path-to-conda-environment>/bin/
2. export RSTUDIO_WHICH_R=<path-to-conda-environment>/lib/R/bin/R
```

Can I use `conda install rstudio`?

There is a rstudio conda package, but this hasn't been updated since early 2020.

Due to the age of this version, ARCC has not tried to use this old version.

Create an R Kernel for a Jupyter Notebook

Goal: Detail how to update an R Conda environment so it can be used as a kernel within ARCC's Jupyter service.

General Process

The general process involves updating the conda environment to include kernel related packages, and then configuring the kernel spec to allow it to be picked up by the Jupyter service.

1. Activate you R Conda Environment.
2. Start R.
3. Install the `IRkernel` package.
4. Exit R.
5. Deactivate your Conda environment.

6. Copy the created `kernelspec` into your home `.local/share/jupyter/kernels/` folder.
7. Update the kernel.json.

Install the IRkernel package

Activate you R Conda Environment and start R.

```
[salexan5@mblog2 ~]$ module load miniconda3/24.3.0
[salexan5@mblog2 ~]$ conda activate /project/arcc/software/conda-
envs/r_4.3.3_env/
(/project/arcc/software/conda-envs/r_4.3.3_env) [salexan5@mblog2 ~]$ R
> install.packages('IRkernel')
...
also installing the dependencies ‘fastmap’, ‘fansi’, ‘utf8’, ‘htmltools’,
‘pillar’, ‘base64enc’, ‘repr’, ‘evaluate’, ‘IRdisplay’, ‘pbdZMQ’, ‘crayon’,
‘jsonlite’, ‘uuid’, ‘digest’
...
* DONE (IRkernel)
```

Created `kernelspec` folder

Installing the `IRkernel` package will create a kernel spec that we can use.

This can be found under the Conda environment location, under:
`lib/R/library/IRkernel/kernelspec/`

```
[salexan5@mblog2 ~]$ ls /project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library/IRkernel/kernelspec/
kernel.js  kernel.json  logo-64x64.png  logo-svg.svg
[salexan5@mblog2 ~]$ cat /project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library/IRkernel/kernelspec/kernel.json
{"argv": ["R", "--slave", "-e", "IRkernel::main()", "--args",
"{connection_file}"],
 "display_name":"R",
 "language":"R"
}
```

Configure Your Jupyter Environment

If you haven’t used the Jupyter service, then you might not have, and thus will need to create the following folders:

```
.local/share/jupyter/kernels/
```

Copy the created `kernelspec` into your home:

```
[salexan5@mblog2 ~]$ cd .local/share/jupyter/kernels/
[salexan5@mblog2 kernel]$ mkdir r_4.3.3
[salexan5@mblog2 kernel]$ cd r_4.3.3
[salexan5@mblog2 r_4.3.3]$ cp /project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library/IRkernel/kernelspec/* .
```

Update the `kernel.json` file to:

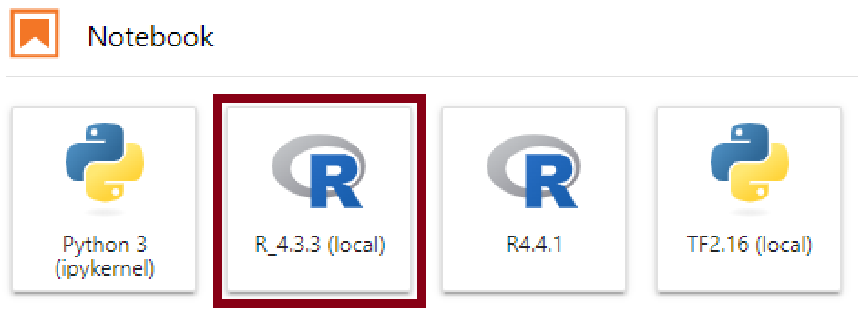
- point to the Conda environment's version of R.
- give this kernel a unique display name.

```
[salexan5@mblog2 r_4.3.3]$ cat kernel.json
{"argv": ["/project/arcc/software/conda-envs/r_4.3.3_env/bin/R", "--slave",
"-e", "IRkernel::main()", "--args", "{connection_file}"],
 "display_name":"R_4.3.3 (local)",
 "language":"R"
}
```

Start Jupyter

From OnDemand start a Jupyter session.

Notice how the newly configured kernel is available.



Within a Notebook

Within a cell try:

```
.libPaths()
# Cell Output:
'/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env/lib/R/library'
write.table(installed.packages()[,c(1,2,3:4)])
```

```
# Cell Output:
"Package" "LibPath" "Version" "Priority"
"base" "base" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
...
"vctrs" "vctrs" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "0.6.5" NA
```

Note the library path is that of the Conda environment: `/project/arcc/software/conda-envs/r_4.3.3_env/lib/R/library`

R Environments and Reproducibility

Goal: Introduce ideas and practices to assist in managing the reproducibility of R environments.

What Packages Do I Have Installed?

First step is knowing what your environment is using, and where these packages are installed:

Remember: to use `.libPaths()`

```
[]$ module load gcc/13.2.0 r/4.4.0
[]$ R
R version 4.4.0 (2024-04-24) -- "Puppy Cup"
...
> .libPaths()
[1] "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-library/4.4"
[2] "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
> quit()

[salexan5@mblog1 ~]$ ls /apps/u/spack/gcc/13.2.0/r/4.4.0-
pvzi4gp/rlib/R/library
base  compiler  datasets  graphics  grDevices  grid  methods  parallel
splines  stats  stats4  tcltk  tools  translations  utils

[salexan5@mblog1 ~]$ ls /cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4
```

```
class     cli    DBI    e1071  generics  KernSmooth  magrittr  pillar
proxy  R6    rlang  sf       stringr  tidyr       units  vctrs  wk
classInt  cpp11  dplyr  fansi  glue      lifecycle   MASS      pkgconfig
purrr  Rcpp  s2     stringi  tibble   tidyselect  utf8   withr  XML
```

Anything ARCC has installed will not be updated. We will create a new version of base R.

Track the R Packages and Versions you have Installed

How can I track the versions of R packages installed? Using plain R:

```
[salexan5@mblog1 ~]$ R
R version 4.4.0 (2024-04-24) -- "Puppy Cup"
...
> write.table(installed.packages()[,c(1,2,3:4)])
"Package" "LibPath" "Version" "Priority"
"class" "class" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "7.3-22" "recommended"
...
"sf" "sf" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-library/4.4"
"1.0-16" NA
"stringi" "stringi" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.8.4" NA
"stringr" "stringr" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "1.5.1" NA
"tibble" "tibble" "/cluster/medbow/home/salexan5/R/x86_64-pc-linux-gnu-
library/4.4" "3.2.1" NA
...
"tools" "tools" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
"utils" "utils" "/apps/u/spack/gcc/13.2.0/r/4.4.0-pvzi4gp/rlib/R/library"
"4.4.0" "base"
```

Conda Export and R Packages

The `conda list` command (within an activated Conda environment) will only list the packages you've installed using `conda install`.

It does not track/list anything you've installed, from within R, using `install.packages()`.

Using `conda env export`/`conda env create` create an incomplete environment.

```
[salexan5@mblog2 ~]$ module load miniconda3/24.3.0
conda activate /cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env
```

```
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 ~]$
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 ~]$ conda list
# packages in environment at /cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env:
#
# Name                    Version                   Build  Channel
...
r-base                    4.3.3                he2d9a6e_3    conda-forge
r-stringi                 1.8.4              r43hbd1cc82_0    conda-forge
...
```

Full conda list

```
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 ~]$ conda list
# packages in environment at /cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                 conda_forge    conda-forge
_openmp_mutex             4.5                       2_gnu    conda-forge
_r-mutex                  1.0.1               anacondar_1    conda-forge
binutils_impl_linux-64    2.40                 ha1999f0_7    conda-forge
bwidget                   1.9.14               ha770c72_1    conda-forge
bzip2                     1.0.8                hd590300_5    conda-forge
c-ares                    1.28.1               hd590300_0    conda-forge
ca-certificates           2024.6.2             hbcca054_0    conda-forge
cairo                     1.18.0               hbb29018_2    conda-forge
curl                      8.8.0                he654da7_0    conda-forge
expat                     2.6.2                h59595ed_0    conda-forge
font-ttf-dejavu-sans-mono 2.37                 hab24e00_0    conda-forge
font-ttf-inconsolata      3.000                h77eed37_0    conda-forge
font-ttf-source-code-pro  2.038                h77eed37_0    conda-forge
font-ttf-ubuntu           0.83                 h77eed37_2    conda-forge
fontconfig                2.14.2               h14ed4e7_0    conda-forge
fonts-conda-ecosystem     1                             0    conda-forge
fonts-conda-forge         1                             0    conda-forge
freetype                  2.12.1               h267a509_2    conda-forge
fribidi                   1.0.10               h36c2ea0_0    conda-forge
gcc_impl_linux-64         13.2.0              h9eb54c0_13    conda-forge
gfortran_impl_linux-64    13.2.0              h9efe08d_13    conda-forge
graphite2                 1.3.13            h59595ed_1003    conda-forge
gsl                       2.7                  he838d99_0    conda-forge
gxx_impl_linux-64         13.2.0              h2a599c4_13    conda-forge
harfbuzz                  8.5.0                hfac3d4d_0    conda-forge
icu                       73.2                 h59595ed_0    conda-forge
kernel-headers_linux-64   2.6.32              he073ed8_17    conda-forge
keyutils                  1.6.1                h166bdaf_0    conda-forge
krb5                      1.21.3               h659f571_0    conda-forge
ld_impl_linux-64          2.40                 hf3520f5_7    conda-forge
lerc                      4.0.0                h27087fc_0    conda-forge
libblas                   3.9.0           22_linux64_openblas    conda-forge
libcblas                  3.9.0           22_linux64_openblas    conda-forge
libcurl                   8.8.0                hca28451_0    conda-forge
libdeflate                1.20                 hd590300_0    conda-forge
libedit                   3.1.20191231         he28a2e2_2    conda-forge
```

```
libev                      4.33                 hd590300_2    conda-forge
libexpat                   2.6.2                h59595ed_0    conda-forge
libffi                     3.4.2                h7f98852_5    conda-forge
libgcc-devel_linux-64      13.2.0             hdb50d1a_113    conda-forge
libgcc-ng                  13.2.0              h77fa898_13    conda-forge
libgfortran-ng             13.2.0              h69a702a_13    conda-forge
libgfortran5               13.2.0              h3d2ce59_13    conda-forge
libglib                    2.80.2               h8a4344b_1    conda-forge
libgomp                    13.2.0              h77fa898_13    conda-forge
libiconv                   1.17                 hd590300_2    conda-forge
libjpeg-turbo              3.0.0                hd590300_1    conda-forge
liblapack                  3.9.0           22_linux64_openblas    conda-forge
libnghttp2                 1.58.0               h47da74e_1    conda-forge
libopenblas                0.3.27          pthreads_h413a1c8_0    conda-forge
libpng                     1.6.43               h2797004_0    conda-forge
libsanitizer               13.2.0              h6ddb7a1_13    conda-forge
libssh2                    1.11.0               h0841786_0    conda-forge
libstdcxx-devel_linux-64   13.2.0             hdb50d1a_113    conda-forge
libstdcxx-ng               13.2.0              hc0a3c3a_13    conda-forge
libtiff                    4.6.0                h1dd3fc0_3    conda-forge
libuuid                    2.38.1               h0b41bf4_0    conda-forge
libwebp-base               1.4.0                hd590300_0    conda-forge
libxcb                     1.16                 hd590300_0    conda-forge
libzlib                    1.3.1                h4ab18f5_1    conda-forge
make                       4.3                  hd18ef5c_1    conda-forge
ncurses                    6.5                  h59595ed_0    conda-forge
openssl                    3.3.1                h4ab18f5_1    conda-forge
pango                      1.54.0               h84a9a3c_0    conda-forge
pcre2                      10.44                h0f59acf_0    conda-forge
pixman                     0.43.2               h59595ed_0    conda-forge
pthread-stubs              0.4               h36c2ea0_1001    conda-forge
r-base                     4.3.3                he2d9a6e_3    conda-forge
r-stringi                  1.8.4             r43hbd1cc82_0    conda-forge
readline                   8.2                  h8228510_1    conda-forge
sed                        4.8                  he412f7d_0    conda-forge
sysroot_linux-64           2.12                he073ed8_17    conda-forge
tk                         8.6.13          noxft_h4845f30_101    conda-forge
tktable                    2.10                 h8bc8fbc_6    conda-forge
xorg-kbproto               1.0.7             h7f98852_1002    conda-forge
xorg-libice                1.1.1                hd590300_0    conda-forge
xorg-libsm                 1.2.4                h7391055_0    conda-forge
xorg-libx11                1.8.9                hb711507_1    conda-forge
xorg-libxau                1.0.11               hd590300_0    conda-forge
xorg-libxdmcp              1.1.3                h7f98852_0    conda-forge
xorg-libxext               1.3.4                h0b41bf4_2    conda-forge
xorg-libxrender            0.9.11               hd590300_0    conda-forge
xorg-libxt                 1.3.0                hd590300_1    conda-forge
xorg-renderproto           0.11.1            h7f98852_1002    conda-forge
xorg-xextproto             7.3.0             h0b41bf4_1003    conda-forge
xorg-xproto                7.0.31            h7f98852_1007    conda-forge
xz                         5.2.6                h166bdaf_0    conda-forge
zlib                       1.3.1                h4ab18f5_1    conda-forge
zstd                       1.5.6                ha6fb4c9_0    conda-forge
(/cluster/medbow/project/arcc/software/conda-envs/r_4.3.3_env)
[salexan5@mblog2 ~]$ R
R version 4.3.3 (2024-02-29) -- "Angel Food Cake"
...
```

```
> .libPaths()
[1] "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library"
> write.table(installed.packages()[,c(1,2,3:4)])
```

R Packages and Versions:

```
> write.table(installed.packages()[,c(1,2,3:4)])
"Package" "LibPath" "Version" "Priority"
"base" "base" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"base64enc" "base64enc" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "0.1-3" NA
"cli" "cli" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "3.6.3" NA
"compiler" "compiler" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"crayon" "crayon" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.5.3" NA
"datasets" "datasets" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"digest" "digest" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "0.6.36" NA
"evaluate" "evaluate" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "0.24.0" NA
"fansi" "fansi" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.0.6" NA
"fastmap" "fastmap" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.2.0" NA
"glue" "glue" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.7.0" NA
"graphics" "graphics" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"grDevices" "grDevices" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"grid" "grid" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"htmltools" "htmltools" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "0.5.8.1" NA
"IRdisplay" "IRdisplay" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.1" NA
"IRkernel" "IRkernel" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.3.2" NA
"jsonlite" "jsonlite" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.8.8" NA
"lifecycle" "lifecycle" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.0.4" NA
"methods" "methods" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"parallel" "parallel" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"pbdZMQ" "pbdZMQ" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "0.3-11" NA
"pillar" "pillar" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.9.0" NA
"repr" "repr" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.1.7" NA
```

```
"rlang" "rlang" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.1.4" NA
"splines" "splines" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"stats" "stats" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"stats4" "stats4" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"stringi" "stringi" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.8.4" NA
"tcltk" "tcltk" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"tools" "tools" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"utf8" "utf8" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.2.4" NA
"utils" "utils" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "4.3.3" "base"
"uuid" "uuid" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "1.2-0" NA
"vctrs" "vctrs" "/cluster/medbow/project/arcc/software/conda-
envs/r_4.3.3_env/lib/R/library" "0.6.5" NA
() []$ ls /project/arcc/software/conda-envs/r_4.3.3_env/lib/R/library/
base       compiler  digest    fastmap   grDevices  IRdisplay  lifecycle
pbdZMQ  rlang    stats4   tools         utils
base64enc  crayon    evaluate  glue      grid       IRkernel   methods
pillar  splines  stringi  translations  uuid
cli        datasets  fansi     graphics  htmltools  jsonlite   parallel
repr    stats    tcltk    utf8          vctrs
```

Track the Building of Your Environments

You will need to use a combination of:

- System: `module load r/<version>`
- R: `.libPaths()`
- R: `install.packages()`
- Conda: `conda list`/`conda env export`/`conda env create`

To record and track how your environment is made up.

Be aware that updating a package might update all it’s dependencies.

The order you install packages might also make a difference.

Install R Packages with a Specific Version

R’s base `install.packages()` only allows you to install a specific version of a package when you’ve downloaded the source.

The `conda install` does allow you to define a specific version.

There are a number of R packages to assist you:

- remotes: R Package Installation from Remote Repositories, Including 'GitHub'
- versions: Query and Install Specific Versions of Packages on CRAN
- devtools: Tools to Make Developing R Packages Easier

Suggested Best Practices

- For specific projects/research focuses, create specific libraries and or conda environments (with everything installed within that conda environment) to localize used packages/versions.
- Regularly track/update what packages you’re using (install.packages / conda install r-<package-name>) and their versions.
- Be mindful of dependencies that a package additional installs.
- Be mindful when prompted whether you want to update dependencies or not.
- Avoid trying to have a behemoth of a single environment - consider have a number of independent environments/libraries that can be more easily managed and shared along a workflow/pipeline.

Parallel R: Introduction

Goal: Introduction some high-level aspects of using R in parallel relating to the cluster.

In the same spirit as this is not a course on learning the R language, this is not a section on developing parallelized code with any of the 10s of parallel related packages.

Instead it will detail some aspects to consider regards using our cluster.

Parallel Programming with R

The are 10s of potential packages that could be used, as a starting point we'd direct your to here: CRAN Task View: High-Performance and Parallel Computing with R.

One thing to consider with respect to what package you wish to explore is whether it provides multi-node functionality (such as Rmpi) or just multicore (parallel) on a single compute node, and/or *cluster features*.

Remember: Just asking for multiple nodes (and GPUs) won't actually make your code run faster unless the underlying package can actually utilize them.

R `parallel` Package: Overview

The `parallel` package is now part of the core R installation and is a *base* package.

It does not need to be installed.

Building Rmpi from Source

If you wish to try to install `Rmpi`, you should use the latest implementation of `OpenMPI` on the cluster to build against.

Multicore: Detecting Cores

Typically, using `parallel::detectCores()` to detect the number of available cores on a cluster node is a slight *red herring*. This returns the *entire total* number of cores of the node your job is allocated and not the actual number of cores you requested/allocated.

For example, if you're `sbatch` script defines the following,

```
#SBATCH --nodes=1
#SBATCH --cpus-per-task=8
```

and you're allocated a standard compute node that has 32 cores, `parallel::detectCores()` will return a value of 32 and not 8 which is what you requested!
This will probably lead to unexpected results/failures when you try and run a function expecting 32 cores when only 8 are actually available.
To remove this problem you can use, and need to pass into your R script, the value of the `$SLURM_JOB_CPUS_PER_NODE` slurm environment variable.

Detect Cores Example

r_multicore.R

```
args <- commandArgs(trailingOnly = TRUE)
if (!is.na(args[1])) {
  num_of_cores <- args[1]
  print(paste0("Num of Cores: ", num_of_cores))
}

print(paste0("detectCores: ", parallel::detectCores()))

options(mc.cores = num_of_cores)
print(paste0("mc.cores: ", getOption("mc.cores", 1L)))
```

```
# Create an interactive session that uses 8 cores:
[salexan5@mblog2 ~]$ salloc -A arcc -t 10:00 -c 8
salloc: Granted job allocation 861904
salloc: Nodes mbcpu-001 are ready for job
[salexan5@mbcpu-001 ~]$ module load gcc/13.2.0 r/4.4.0

# Check the slurm environment variable: SLURM_JOB_CPUS_PER_NODE
[salexan5@mbcpu-001 ~]$ echo $SLURM_JOB_CPUS_PER_NODE
8

# What does R detect?
[salexan5@mbcpu-001 ~]$ Rscript r_multicore.R $SLURM_JOB_CPUS_PER_NODE
[1] "Num of Cores: 8"
[1] "detectCores: 96"
[1] "mc.cores: 8"
```

Using R/RStudio on the Cluster: Summary

Goal: Summarize the concepts covered across the workshop.

Summary

Looked at:

- Where R (loaded from a module) installed packages with respect to the version of R.
- Where an R environment built with Conda installs packages.
- Inspecting the paths returned from `.libPaths()` to understand and modify where packages are installed.
- Using R's `install.packages()` command and that additional modules/libraries might need to be loaded into your environment for packages to successfully be installed.
- Using Conda's `install` to install packages.
    - Similarly, you might need to `conda install` additional libraries.
- How R libraries can be shared across a user's environments and potentially how updating a package version can impact these environments that share the library and packages.
- How to create an R library that can be shared by users across a *project*.
- How to use RStudio via OnDemand and an Interactive Desktop.
- How to configure a session to use an R environment within a Conda environment.
- How to take an existing R Conda Environment and update it to use as a kernel within Juypter via OnDemand.
- The more complicated an environment, the more packages you're trying to install, the more likely you're of hitting dependency hell.
    - Consider having a number of smaller environments.
- The notion of reproducibility of environments.
- Some high-level aspects to consider when parallel programming within R on the cluster.

Use the following link to provide feedback on this training:
https://forms.gle/da11o2nGjeZtE7DLA or use the QR code below.