rOpenSci Community Call - {targets} in Action

This doc - https://bit.ly/targets-in-action - is a place for collaborative note-taking.
It includes:

- Agenda
- Attendees list. Please add yourself
- Questions from attendees. All are encouraged to ask and answer in this doc.
- Resources. Add your favorites.
- Transcript of the call will be added afterward.

Information on today's topic, with speaker bios: https://ropensci.org/commcalls/jan2023-targets/

This **call will be recorded** and posted along with any other resources at https://ropensci.org/commcalls.

Agenda

1. Welcome, Yani Bellini Saibene (2 min)
2. Eric Scott: *Harnessing HPC power with {targets}* (15 min)
    - Slides: https://cct-datascience.quarto.pub/harnessing-hpc-power-with-targets/
3. Joel Nitta: *Using {targets} for bioinformatics pipelines* (15 min)
    - Slides: https://joelnitta.github.io/comm-call-bioinfo-targets/
    - Code: https://github.com/joelnitta/targets_vcf_example
    - Blog post: https://www.joelnitta.com/posts/2021-11-16_r-bioinfo-flow/
4. Will Landau: *Debugging {targets} pipelines.* (15 min)
    - Slides: https://wlandau.github.io/targets-debug/
    - Code: https://github.com/wlandau/targets-debug
5. Q & A moderated by Yani (15 min)
6. Wrap-up, Yani (1 min)

You all have expertise to share! Add your questions by typing them below, and share your thoughts by answering others' questions. Together we can make this a rich resource for all of us.

In attendance (please add yourself here):

- *Name; Affiliation; Country*
- Yani, rOpenSci, Argentina

- Eric Scott, University of Arizona, USA
- Joel Nitta, University of Tokyo, Japan
- Federica Gazzelloni, Italy
- Dragos Moldovan-Grünfeld, NPL Markets, UK
- Steffi LaZerte, rOpenSci, Canada
- Will Landau: Eli LIlly and Company, USA
- Micha Silver, Ben Gurion Univ., Israel
- Jacob Krol; Colorado University; USA
- Leo Timmermans, Netherlands
- Brandon Farr; Congress Asset Management; USA
- Andrés Castro Socolich; none; Perú
- Jacob Hannan, MCR Performance Solutions, USA
- David Duncan, TransUnion, USA
- Monica Alonso, Buenos Aires, ARGENTINA
- Kari Norman, University of Montreal, Canada
- Alec Robitaille, Memorial University, Canada
- Salix Dubois, Université du Québec à Montréal, Canada
- Nathan Layman, EcoHealth Alliance, USA
- David Schoch, GESIS, Germany
- Jared Norman; University of Cape Town/MASHA; South Africa
- Russell Death Massey University Palmerston North New Zealand
- Jeffrey Stevens, University of Nebraska-Lincoln, USA
- Pablo Tiscornia, Ciudad de México, México
- Julian Tagell, Tadge Analytics, Australia
- James Lackey, KPMG US & University of North Texas, USA
- Katherine Hébert, Université de Sherbrooke, Canada
- Cynthia Huang, Monash University, Australia
- Tom Briggs, George Mason University, USA
- Elio Campitelli, University of Buenos Aires, Argentina.
- Travis Hinkelman, Environmental Science Associates, USA
- Nicholas Potter, USDA Economics Research Service
- Pierre L'Helgoualc'h, Monash University, Australia
- Erin Steiner, NOAA, USA
- Carlos Zambrana Torrelio, George Mason University, USA
- Lev Gorenstein, Purdue University, USA
- Yutong Song, Monash University, Australia
- Craig Robert Shenton, NHS England, UK
- Steve Lianoglou, HI·Bio, USA
- Gracielle Higino; University of British Columbia; Canada
- Emma Mendelsohn, EcoHealth Alliance, USA
- James Azam, Epiverse Initiative (London School of Hygiene and Tropical Medicine), UK
- Mar Barrio Luna; Okuant; Spain
- Andrew Clarke; Australian Red Cross Lifeblood; Australia

- Simon Rolph, UK Centre for Ecology and Hydrology
- Hebah Bukhari, Zpryme, USA
- Michal Kinel, Okuant, Madrid, Spain
- Noam Ross, rOpenSci/EcoHealth Alliance, New York, NY, USA
- Sergio Olmos, Sanofi, Barcelona, Spain
- Nistara Randhawa, UC Davis, CA, USA
- Henrik Bengtsson, UC San Francisco, CA, USA

Questions from attendees

Add your questions by typing them below, and share your thoughts by **answering others' questions**. Together we can make this a rich resource for all of us.

- *Name (optional): question(s)*
- Joel Nitta: Are there any particular targets scenarios (e.g., hundreds of parallel targets, etc) where tar_make_clustermq() or tar_make_future is more performant?
    - Eric: The clustermq backend is probably faster in most cases. The more targets/workers there are the bigger difference it makes.
    - Will: For the clusters I have worked with, I agree with Eric. Overhead from clustermq is minimal if the data isn't to big. For large targets, if you are not using the clustermq SSH connector, you may have better luck with retrieval = "worker" and storage = "worker" in tar_target(). That way, the workers read and write the data to/from disk and do not send it over ZeroMQ sockets. `targets` is designed to handle the lag you may experience on network file systems on clusters (by watching checksums to make sure files are synced among nodes). (More details on performance tricks are at https://books.ropensci.org/targets/performance.html). The future.batchtools overhead situation may be get better in the near future: https://github.com/HenrikBengtsson/future.batchtools/issues/65#issuecomment-1405797594.
    - Will: one more thing: with persistent clustermq workers, it is important to avoid idle time if you can. Sometimes if you have a massively parallel part of the pipeline and it funnels into a small number of quick downstream targets, those downstream targets can use `deployment = "main"` to run on the main controlling R process instead of on a worker / HPC job. That lets `targets` start terminating `clustermq` workers one-by-one when it knows those workers will no longer be needed.
- Nathan Layman: Are there sequential target branching patterns? Something like accumulate? There would still be a benefit of skipping steps that have already successfully run. An example might be a random walk where each step is captured as a

target. Sorry, more information. I'm asking about sequential not parallel branching. Where the next branch operates on the results from the last branch run.
    - Will: There are some small target factories in {tarchetypes} like tar_file_read() which define more than one target in sequence. Currently these are for isolated special cases. The more general branching patterns are meant to be parallel.
- Nicholas Potter: What are best practices for using targets in a large data scenario? From what I can tell, target outputs are stored in the targets cache, which can make working with multi GB difficult.
    - Will: I would first look at the available storage formats: https://docs.ropensci.org/targets/reference/tar_target.html#storage-formats. The list includes formats like "feather" and "parquet" for efficient storage and retrieval of large data frames. Otherwise, if your input files are large, it may be best to use tar_target(format = "file") or tar_file() to track those files and then ensure the downstream targets only read the relevant rows/subsets.
- Andrés: Are there patterns for using targets with RDMS' (e.g. Postgresql)?
    - Erin: I create target objects that are dedicated to just pulling data from databases and then use tar_cue() to manually determine when to do an updated pull
    - Noam: We do something similar. We usually have disconnected parts of the targets graph for processes that populate/update the database and processes that pull from the database.
    - Andres: But how to avoid pulling and storing data locally? Any pattern using SHAsums to just pull data when is changed to render an output but not saving the data locally?
    - Will: Noam was just telling me the other day about his workaround with cues. The `cue` argument of tar_target() controls the conditions under which an existing target reruns, and it accepts an object returned from tar_cue(). For a target that pulls from a database, you could make the `mode` argument of tar_cue() depend on the hash of the table you want, or an environment variable you set. Alternatively, you could set `cue = tar_cue_age(...)` to automatically invalidate a target when a certain amount of time has passed since it last ran.
- Andrés: Are there patterns for using targets for business reporting (e.g. Rmd or qmd reports pulling data from RDBMS')?
    - Erin: There is tar_render() or tar_quarto() that you can add to your targets workflow
    - Andres: Does this mean I have to create a bunch of targets for each plot for example and then just add them to the Rmd/qmd file?
    - Erin: it depends on your workflow, but you can load any target object into rmd/qmd, so my workflow is usually to load in the results objects and then use them however (extract values, summarize into tables, generate figures)
    - Joel: +1 to Erin's answer. I treat the targets plan as the place for "heavy lifting" to do all the statistical analysis, then use the Qmd report to summarize tables and make figures. But if you want to make figures in the targets plan, you may find this blogpost helpful: https://ropensci.org/blog/2022/12/06/save-ggplot2-targets/

    - Will: Excellent, Erin and Joel! I will also add that in a business scenario, if your company has Posit Connect, you can make use of it in a `targets` pipeline. I like to include a tar_file() downstream of the tar_quarto() target which calls rsconnect::writeManifest() on the output HTML file. Then, I commit the HTML report and the manifest.json to an internal repo in my company's GitHub organization, then link that repo to Connect using Git-backed content: https://docs.posit.co/connect/user/git-backed/. Alternatively, you could have a downstream target that calls rsconnect::deployApp() directly.
- David Duncan: is there any plan to allow more flexibility in output types for cached results, particularly for model objects that don't play well with saveRDS? Or maybe what usage patterns work well for storing/exporting model objects?
    - Noam: You can define a custom target format with its own serialization/deserialization method. https://docs.ropensci.org/targets/reference/tar_format.html
        - There are already some custom built-in formats for specific model types, like tensorflow models. https://docs.ropensci.org/targets/reference/tar_target.html#storage-formats
    - Will: +1 to Noam's answer. If one of the built-in formats does not suit your needs, the tar_format() function lets you write your own: https://docs.ropensci.org/targets/reference/tar_format.html.
- Erin: I'd love to hear from anyone that has incorporated "formalized" tests into their targets workflow
    - Joel: I've started using testthat, I think it seems fairly straightforward but haven't done it extensively yet
    - Erin: If you have a repo with an example, could you share it?
    - https://github.com/joelnitta/japan_ferns_spatial_phy/blob/5f79604c6ce923c12b3110f811c57a544265fa16/_targets.R#L80
    - https://github.com/joelnitta/japan_ferns_spatial_phy/blob/main/R/tests.R
    - Will: I have heard reports of success including unit tests in pipelines, including Joel's. For most users, I recommend testing functions outside the pipeline on small data, which adds confidence that your code is working before you launch a potentially time-consuming tar_make(), tar_make_clustermq(), or tar_make_future(). This need not replace validation/assertion checks on the output data objects that the targets generate.
- Elio: What are tricks or things to consider when working with relatively large data to optimize RAM usage (e.g. that objects don't duplicate for each worker)
    - Will: I would recommend memory = "transient" and garbage_collection = TRUE for those cases. More details are at https://books.ropensci.org/targets/performance.html.
- I've been trying to write some targets markdown documents - is there a tar_toggle() function that is used to change a parameter(say the number of replicates) between local prototyping and running on a cluster?

    - Will: tar_render() and tar_quarto() from tarchetypes accept arguments for R Markdown/Quarto parameters (arguments params and execute_params, respectively).
- Noam Ross: What's the current state and near-term expectation for (semi-)transient cloud workers for targets?
    - Will: I am slowly working on a rewrite of https://github.com/wlandau/crew to accomplish this (current branch: https://github.com/wlandau/crew/tree/21). So far I have written an R6 class to manage instances of Redis Server, and I plan to build infrastructure to launch rrq workers (https://github.com/mrc-ide/rrq). I plan to first get this working for SGE clusters, then other kinds of clusters. After that, it will be a journey to get it working on the cloud.
- Simon Rolph: Q for Joel (+ others), what was your motivation for using targets over nextflow?
    - Joel: I have not used nextflow so I can't compare them directly. I use targets because I'm a useR :) but I probably should try nextflow too
    - Eric: same here. Like I said, until recently, I've been *very* wary of stepping outside of my RStudio comfort zone 🙂
    - Will: same, I have only heard about Nextflow, never used it. There are other pipeline tools such as Prefect, Airflow, Kestra, Metaflow, and Mage. Many are listed at https://github.com/pditommaso/awesome-pipeline, and a recent blog post compares some of these: https://chengzhizhao.com/5-fantastic-data-pipeline-orchestration-tools-for-r/. If we include tools from other languages, the ecosystem is enormous. Other pipeline tools are usually written in e.g. Python, and then the R bindings come later, with mixed enthusiasm and mixed results. The strength of `targets` is its commitment to R. `targets` is built entirely in R, it preserves the look and feel of an ordinary R session, and it adopts patterns and workflow principles that are not always part of the culture in other languages. Examples include a focus on functions instead of scripts, R Markdown integration in tarchetypes::tar_render(), and dynamic branching over the dplyr row groups of data frames: https://docs.ropensci.org/tarchetypes/reference/index.html#dynamic-grouped-data-frames.
- Sergio Olmos: Is it possible to use something like the {config} R package to provide different values (say, size of the data or number of models) inside the targets pipeline when the pipeline runs in your local machine and when the pipeline runs in an HPC?
    - Will: You probably can set up a pipeline to use `config`, but `targets` has its own `config`-like system to control the top-level arguments of key functions: https://books.ropensci.org/targets/projects.html#multiple-projects. Basically, you have a `config`-like YAML file in _targets.yaml which you can handle multiple project-level configurations. You can set/get fields in the YAML with tar_config_set() and tar_config_get().

- Jared Norman: Why is R not giving more context of the error initially; we see a vague error without the name of the “offending” function, or line number, or any of the state, etc. Then when we get into the debug view we get some more context.
    - Will: this is one of those layers of encapsulation I mentioned at the beginning. When `targets` throws an error, it calls rlang::abort() with `call` equal to an empty environment: https://github.com/ropensci/targets/blob/main/R/utils_condition.R#L99. The reason is that the code executes in a special temporary environment created for the target instead of the global environment, and there are other similar layers of protection and reproduciblity. So if I were to allow the error call information to be included in the error message, the call information would point to an internal function in `targets` instead of the function you want to know about, and it would be misleading. If you debug with workspaces, you can get a traceback which is more thorough and more accurate: https://github.com/wlandau/targets-debug/blob/a9be0fa7a7f09afaf2bceb3e3009803688119be7/demo_workspace.R#L22

Comments from attendees

- Joel Nitta: vscode has nearly seamless connection with remote sessions (another option for dealing with the local/remote gap) (Eric: yes! I forgot to mention this. I wasn’t aware of this feature until fairly recently, but if I had known about it earlier I might have been convinced to switch to vscode)
- Lev Gorenstein: Running Docker is often not allowed on HPC resources due to the need of elevated privileges. There are alternatives specifically designed for running containers on HPC - Singularity, Apptainer, Podman, Shifter, CharlieCloud. The great part is that all of them can pull down Docker containers into local files, and then execute them locally without Docker.
  E.g. - instead of 'docker run .....' our cluster users can do
  singularity pull fastp-X.Y.Z.sif docker://quay.io/biocontainers/fastp:X.Y.Z # pull down a docker container a local file
  singularity exec fastp-X.Y.Z.sif samtools ..... # and run samtools from the container
  This does marvels for reproducibility (and for sharing artifacts with colleagues)
    - Joel: babelwhale can also run singularity (I personally have not tried it myself though). The syntax should be the same, you just change the container backend:
      ```
      library(babelwhale)
      config <- create_docker_config("singularity")
      set_default_config(config)
      ```
- Henrik Bengtsson: Adding to Lev's comments on Singularity/Apptainer. Highly recommended and accepted as safe and efficient by the HPC community (= much easier to convince sysadms to install than Docker). They also run as the user who calls them,

that is, not as root. They also mount common folders automatically such as the current working directory ($PWD) and user's home ($HOME). For instance, when saving a file "from inside the container" in the current directory, it appears as a regular file owned by the user available on the host file system.

- I really like the help guidance here: https://books.ropensci.org/targets/help.html but I'm wondering if there is a good forum to get friendly feedback / reviews of one's pipeline (without any specific bugs)
    - Joel: rOpenSci slack #pipeline-toolkits or the {targets} discussion forum https://github.com/ropensci/targets/discussions are both good places for friendly discussions that aren't necessarily issues/bugs
    - Will: I agree, and I would add https://discuss.ropensci.org/. And for pipeline reviews etc. at https://github.com/ropensci/targets/discussions and elsewhere, I would really appreciate help with answers. I often find it hard to keep up with the level of activity on the forums (and I wrote https://books.ropensci.org/targets/help.html in part to try to be more efficient).

Resources

Add your favorites below

- {targets} book: https://books.ropensci.org/targets/
- {clustermq} user guide: https://mschubert.github.io/clustermq/articles/userguide.html
- {targets} doc: https://docs.ropensci.org/targets/
- Basic tutorial in Spanish (Español): https://github.com/okuant/targets_tutorial
    - Yani: Gracias!
-

Want to get involved?

Join rOpenSci social coworking and office hours Tuesday, 07 February 2023 17:00 UTC or Tuesday, 07 March 2023 01:00 UTC

Share your perspectives on the Community Calls we're planning

Ideas for upcoming Community Calls are shared as issues in this public repository. Give a 👍🏽 on a topic, share your perspective by commenting on a topic, or open a new issue if your idea doesn't fit in with any others.

Transcript

Yanina:
Hello everyone. I'm Yanina Bellini Saibene, rOpenSci's Community Manager and I welcome you to our community call on **{targets} in Action**.

rOpenSci is a non-profit initiative founded in 2011. We create technical and community infrastructure for open and reproducible research in R. This includes a curated collection of over 300 packages, a software peer review system, the R-universe platform for building, testing, and publishing packages, and documentation and community engagement programs to support scientific R developers.

You can find our newsletter at news.ropensci.org.

We have a Code of Conduct that applies to this Call. You can find it in the footer of our website at ropensci.org along with guidelines for reporting and enforcement.

Today we'll be using a **shared google doc** for collaborative note taking. My colleague Steffi LaZerte will also place the link in the Zoom chat. **Please add your name** and affiliation to the attendees list in the doc.

- I invite you to add your own notes in the doc.
- Have a question? Type it and add your name, the moderator may call on you to unmute and ask. You can add your own responses to other people's questions. Together we can make this a rich resource for all of us.

I'll get us started by **introducing our speakers**

Will Landau is the author of rOpenSci's {targets} R package. He earned his PhD in Statistics from Iowa State University in 2016, and he now works at Eli Lilly and Company, where he develops methods and software for clinical statisticians.

Eric Scott is a Scientific Programmer & Educator at University of Arizona. He has a background in chemical ecology and plant ecology and is a regular contributor to the {webchem} package. He has now attempted to make it easy to set up {targets} to harness the power of HPC from the comfort of an RStudio window at three universities—and succeeded at two!

Joel Nitta is a Project Research Associate at the University of Tokyo. He studies the evolution and ecology of ferns, and is passionate about reproducible data analysis using R.

We will start with Eric presentation, then Joel and finally Will. So, over to you Eric.

Eric (00:03:18):

All right, I will go ahead. Okay, so thanks everyone and yeah, I'm gonna talk about harnessing the sort of power of cluster Computing using the targets package.
So a little bit, you know about my background you heard a little bit already. So, you know, my background isn't as an ecologist and now I work as essentially a research software engineer and data science trainer.
I'm not an HPC professional or expert by any means we have other folks at University of Arizona that provides support for the HPC. So I'm a user of the HPC and you know, I go to those people for help. So that's the disclaimer and then the other thing you need to know about me is that I really like never ever leaving the comfort of Rstudio desktop and that is a motivating. Peace behind what I'm gonna talk about and I think it's actually not that uncommon for researchers to really not want to leave RStudio desktop and you know that something that can make using the HPC at a university or an organization kind of daunting for a lot of people.
So I didn't realize I was going first. I don't have a really intro slide about what targets is but I'll just briefly say it's a really wonderful package for workflow management in R. It can help you to write your your code your analysis code in ways that streamline things and automatically skip the steps that don't need to be rerun when you make changes to your data or your code and it forces you to make parallelization easy.
It forces you to to write your code in ways that that lower the barrier to be running your code and parallel you have modular workflows. And there's also these features of branching that are in the targets package that create these independent steps called targets.
So in this example workflow, there's you know data file that gets red and wrangled into a data frame and then that data frame is used to fit three different models and those three models could all be run.
Dependently, they don't need to wait for the other one to finish and then maybe the results are plotted and those plots are combined into a multi-panel figure something like that.
So there's a bunch of things here. That could be run in parallel. And this is actually pretty easy. If you've set up your workflow using the use underscore targets function in the targets package. Most of it is already going to be automatically set up for you.
It automatically detects what kind of system you're on and then you can use either the tar make cluster mq function or the tar make future function instead of just playing tar make to run your your targets workflow in parallel and both of these take a workers argument. So if we wanted to run that previous workflow with three workers, we could just say workers equals three and then all three of those models would run at the same time in parallel.
And those parallel processes could be processes on your computer on your up or they could be jobs on a Computing cluster. And this is what I think makes it really lowers the barrier and makes this a really powerful way for people to make use of a Computing cluster who may be don't want to spend a lot of time working on the command line.

Um, so I mentioned that there's these two options. There's the tar made cluster mq and tarmac future. I'll briefly describe what the difference is. The book The Target's book describes it in more detail, but essentially cluster mq is an R package. That's the backend for tarmake cluster mq and it uses persistent workers. So you can imagine an office building we've got say four desks workers come in to those four desks sit down. They spend some time getting set up. They get their laptops up. They get their coffee ready. They're computers plugged in and all that and then a manager hands them files when they complete the files. They put them in the donepile. The manager come picks up hands them a new file.

And once they're done, once the there's no more files to hand out. They start to pack up and leave. So there's a one-time cost to set up those workers. But this back end does have a system dependency on a library called zeromq, which you might not want your collaborators to have to install or maybe you can't install it depending on you know your system.
Um, the future backend uses transient workers. And so this if we imagine kind of the same analogy where we've got the four desks here instead though, every worker or every job gets its own worker every Target gets its own worker. So there's a line of people at the door first person gets, you know, the first four people get handed each handed a folder they go down they set up their workspace when they're done. They take they take pack everything up they hand the finished folder in and they leave and then a new worker comes with the next folder, you know and sets up at that desk. So there's a lot more overhead there a lot more times spent those workers, you know, setting up their workspaces and taking them down but there's no additional system dependencies there and that overhead might not matter depending on the workflow that you're running might not make a huge difference, but for me in my experience the kind of workflows, I was running the the cluster mq package was a big improvement was was a lot better so I'm gonna talk about that.And in general terms what it takes to set this up on a cluster. Set this up to work with a cluster at say a university or something. Um first step is to take the basic HPC training at your organization and it's gonna it's gonna come in use later. You're going to be able to copy and paste some things that you learn in that that are specific to your organization or whatever cluster Computer Resources you have at hand.
Then second step is to install that cluster mq R package on your cluster.
Um, and that might not work right away because this zero mq Library might be missing so you might have to open a support ticket to talk to one of your HPC professionals to get that to happen and it would be helpful to send them links to the zero mq website and maybe talk a little bit about why you're doing this and and how this is a really powerful way for our users to use the cluster cluster.
And then once you're get that installed, you can log into your cluster and in a directory somewhere you can launch and run that targets use targets function and this should automatically detect that you're on a cluster and it should figure out what job scheduler your cluster uses in my case. I've always had slurm and I think that one's pretty common, but it might be something else. There's a many other possibilities. It'll create some files including a template slurm.
Template thing and you'll need to edit that and this is where you can go back to that first step where you took the course about how to use the HPC and you might have learned that there are

some things that are required by your particular organization. So for me, it was that I had to tell it this partition hpg default the job wouldn't work if I don't have that in there.
And then these other things I'm gonna leave alone with these double curly braces. These are wild cards that are gonna get filled in by the targets package and the clustermq package when you run stuff in my case. I also had to add some code to load R and a specific version of our and also to load pandoc because I was using it to render some are markdown stuff. So this is code that's running on the HPC.

On the cluster then you'll check that clustermq works on its own go to the package website follow the example, then you can check that tarmake cluster mq works and it's working.

But we're not working comfortably yet. Right. So you might be depending on your organization. You might have to to in order to run our code on the cluster. You might have to be doing this without RStudio. You might have to be working in a terminal window, or maybe you have some options for RStudio, but they're kind of clunky so at University of Florida when I was there the option was this like X11 window running from the terminal and it was very pixelated and laggy and it was just not I was not interested in it.
And then you also have the issue of if you want to do some interactive development and stuff on your computer on your RStudio. Well, then you have to like sync things from one computer to the HPC. So if you run your targets Pipeline on the HPC, it creates this underscore targets folder that has the data store and that's not automatically going to get synced over.
So this is like a common way maybe or one one possible pattern that you would run a target's Pipeline on the hpcs. You do some development on your computer in this targets.or file and and our folder with some scripts. Maybe you'd run kind of do a test run that would produce this data store targets that holds those intermediate steps.
You would think the code up to GitHub but not the data store. You don't want to sync that because it's large objects that are not very get friendly.
So when you pull it down onto the HPC and run it there you have a different separate Target store and they're not they're not synced. So that's not very convenient.
One solution that the targets package provides is that you can put that data store in a in the cloud use cloud computing. So Amazon web services for example, and that way there's just one shared targets data store that's accessed both at your laptop. And when it's run on the HPC and this kind of workflow you still would have to kind of sync your code back and forth. So maybe make some changes to the code on your computer on your our studio.
Sync it to GitHub log into the HPC pull those changes down run it on the HPC and then at least the targets folder would be updated for you. That wasn't really an option for me partly just because you know am AWS requires a credit card and I was very nervous about as a postdoc about putting a credit card in and I hear horror stories about people accidentally getting giant bills for AWS, and I just didn't want to deal with it.
Yeah.So this is what I ended up going with as a postdoc was using this SSH connection option. So this is pretty cool. I think because it lets you work on your computer and rather than syncing the whole project over to the HPC. We're getting a setup so that just individual targets get sent over to the HPC run as jobs, and then the results get returned back to your laptop.

And this is super cool. I think it allows you to develop and run the workflow on your computer. So if you have some results returned, you can just explore them interactively without having to worry about syncing things.
And I think that this workflow is kind of Ideal when maybe only some of your targets really need cluster Computing. You don't need the whole project to be run on the on the cluster. You just have one step where you've got, you know, 10 long running models that you all want to be running in parallel, you know on separate cores or something like that.
It's great when your targets don't run too long because you do need to still have the the r session on your laptop like open and running. So it's not great if your steps are taking weeks to run but if they take you know a day, that's probably fine.
And it's also useful if there's really no other alternative way to comfortably use our studio on your cluster.
So to set that up, there are some additional steps, but they're not too bad. Essentially, you're going to copy that slurm template from before onto the cluster.
And then on the cluster you're going to create a DOT our profile where you have some options that point to where that template is. So this is pointing to that template.
And saying that the scheduler is slurm or whatever it is on your system.And then on your laptop in your targets.or file where you're developing your pipeline you use the SSH scheduler.
You give it your your username and host or however you SSH however you log into your cluster.
I found that it was pretty important to increase this SSH timeout.
Above the default because one of the bottlenecks is getting the data back and forth from the HPC from the cluster.
And then this specifies where a log file gets saved. Now. The other thing to remember is you have to install our packages all the r package you're using using on the cluster 2 so that those targets can get run over there. So it's not too bad to set it up but it is a journey and I'll tell you a little bit about some of the lessons I learned along the way so University of Florida, where as a postdoc. I found that the transfer of our objects back and forth from the HPC with that is the biggest bottleneck for that SSH connector.
And so that involved thinking about how I could kind of slim down slim down model objects before that. They got sort of saved as a Target and then transferred back to my computer and also like increasing that time out.
I was also surprised there was a point where Universe University of Florida started using requiring two-factor authentication for that for the cluster and I was sure that was going to break everything that I had done but it really didn't all I do is I run tar make cluster mq on my computer hit go I get a push notification on my phone. I authenticate and it runs.
So then I tried to set up the same thing at Tufts University where I was an affiliate and there I just couldn't get somebody to email me back about getting zero mq installed on the cluster. That was the biggest barrier for me there. I could have used the the future backend because it doesn't have that system Library requirement but the overhead of setting up those workers waiting for those slurm jobs to get allotted to me or whatever was was really too much to be helpful because I wasn't I had a lot of short running targets basically and then where I'm at now a University of Arizona and has proved basically impossible to set up this SSH connector.

Part of the reason. Why is that SSH connector does require an R session to be running on the login node. And this is usually frowned upon at a cluster computers at universities, especially but it's not even possible at University of Arizona. You cannot even start R on a login node. You have to request and interactive session and I haven't figured out how to get that to work.
But fortunately there's this cool alternative. We have something called open on demand our studio server where you can launch a RStudio server window in your browser browser window using however many cores and however much memory you want per core on the cluster. So it's running on the cluster and it works really well. The only thing that's a little funky is when you use that used targets function it thinks you're on a cluster computer which you are and so it sets everything up to be for slurm jobs, but that actually doesn't work and you have to change it to be running as a multi-core or actually it might be multiprocess. I can't remember but you have to set it up as if it were running on your laptop because that's what works.

And then the last step of all this setup is to write it down somewhere and share that with other people at your organization. So I think it's a great idea to make a template GitHub repo with some setup instructions and the readme and like all those template files and stuff kind of already filled out as much as possible and then to distribute that and tell tell somebody some of the HPC experts that you're your organization about it so that other people who are using targets can can find out about it. And I just have some examples here from University of Florida and I'm still working on one of these for University of Arizona. And one of the very cool things that I stole from I think Noam Ross is the ability to actually display your workflow right in the readme in this like expandable thing which is which is super cool and pretty easy to do.
That's all I have to say.

Yani: Thank you. Thank you Eric. There is some questions on the Google Doc if you want to check while Joel, now Joel will take the mic.
Eric: Yeah, we want to do questions now.

Yani: No, wait, if you want to go look and then at end we are going to answer some of them but if you already wants to because there are several so I don't know we are going to have time to answer all.

Eric: Oh, yeah, and then speaking anymore. I'm also hoping that Will will answer most of the questions I get it.

Joel: Okay great. Can you hear me? Okay. Well, I'm super excited to be on this panel. So thank you for inviting me especially with Eric and Will the none other than the creator of targets. So I'll be talking about using targets for bioinformatics Pipelines.
Okay. Is that moving? Okay, wait a second. Sorry. I press the wrong button on Zoom. Okay. There we go. Yeah, so yes, my name is Joel Nitta. I'm a project research associate at University of Tokyo, and I study the ecology and evolution of ferns, which I'm not talking about today, but if anyone ever wants to talk about it, I'd be more than happy.

These are a couple of resources and and do please check the the notes that have been shared for this talk that have the link to my slides because my slides have a lot of links in them. So I think it'll be helpful for you to click on those as you go through and these are just a couple of additional resources on the same topic. This is actually based on a blog post. It's now a little bit old. So the information you're getting here is going to be more up to date but the same ideas apply blog posts that I wrote and then also there's a demo code repo to go along with it.
Okay, so what do I mean by bioinformatics? Well bioinformatics typically involve a scenario kind of like this where we have a lot where we start with some raw data input and as a file and then it goes through this long series of getting put into different programs various external programs and getting intermediate files spit out and then going into more programs and then we finally get up we end up with the final result at the end. So this is you know, a pipeline much like a target's pipeline, but the big difference between your typical targets pipeline is that it's involving all of these external programs and lots of files, right? So for the purposes of this talk, and of course that's typically done with biological data like DNA sequences and stuff, but it doesn't have to be so for the purposes of this talk when I say Automatics, I mean any analysis that involves calling lots of external programs from our and using external files, so if that sounds like kind of familiar to you, then you might find what I'm presenting here useful, okay?
An important goal of this sort of analysis is that it should be reproducible. Okay, so that either your future self or another person can come back and do it and get the same result.
This is a broad overview of how I set this up and I'm gonna go into each of these aspects in more detail and further slides, but just to kind of what your appetite the way I do. This is I use Conda for maintaining my overall environment and inside that I'm running R. And then of course, we have our package targets to maintain the workflow and from targets. I'm calling Docker using
wrapper functions to call each of those external programs.
So that sounds like a lot but I'm gonna break it down in the next slide. So let's as I mentioned one major difference between this sort of setup versus your quote unquote typical Target setup is that we're using a lot of external files. So that has a couple implications for one thing instead of using the the normal tar Target function to Define your targets. A lot of times. We're going to be using tar file because we're going to be saving the results to external files.
Another aspect is that you have to decide where you're gonna put all those files, right and I recommend them putting in a sub directory of the target's cache called user and this is a special folder that will is set up that allows you to just sort of check any files in there that you need for your workflow.
Now a bonus of doing this is that you can then use the git targets package also written by Will to actually take snapshots of the state of your Target's cash. And so what that means is that it allows you if you say you roll back to some particular state of your targets workflow later. You make some change you roll it back. Well, maybe that might be a simple change in terms of the code, but it could have called some program that took like two days to run.
So if you've taken a snapshot of it using git targets you could then also check out the state of your Target's cash immediately and not have to rerun everything.
So there's a few more recommendations I have for using that. User folder to store your data. I find it useful to then have some subdirectories in there for for organization. So I typically put my

raw data files into folder called data, and of course, you should never modify those after they've been finalized for for use in your analysis and then the other two folders that I typically use I call intermediates and results. So intermediates holds all those intermediate steps and we'll probably have some Boulders in there to organize those more and then of course results is a final results and those can all be overwritten and you know deleted as necessary.
Um, so a big part of as most of you know, if you've worked with targets at all a big part of using targets is writing your custom functions. And for this sort of pipeline, we're going to be writing a lot of wrapper functions that call external programs from our and base R comes with a couple of functions to do that. The simplest one is called system and system. Basically, you just put give it a character string that is what you would be typing at the terminal so it's very easy to use but it's pretty Limited in some aspects.
There's another function called system 2 which as the name would imply is a little bit more, you know, it's the next version up from system. It's a little more complicated use but it gives you some more control over what you're doing. And then one that I actually prefer is a separate package called process X and that's the process X run function. This has some great features. For example, it can allow for asynchronous processes. You can set up multiple background processes one thing that's really useful is that you can specify a different working directory than your current working directory where you're running R and oftentimes these programs that you run might make assumptions about the working directory and you would normally have to change that. But if any of you have watched Jenny Bryan's talks, we're ready for blog posts. You'll know that changing your working directory in our session was set WD is kind of a No-No it can make things really confusing for if you if for yourself or others if you try to reproduce that code, so I would generally, you know not do that. And again, that's why process X run is really really handy. It can also you can tell it to put the Standard air and standard out into different files if you want and other features, so the way that you use prospects run basically it takes the the two basic arguments. The first one is the command to run and then the second one is a character Vector of Arguments to that command. So for example, if we're going to run the ls command to list the files in our working directory, we would say LS and then give it a character Vector with some options. So like for example the dash Al Dash H, which is would be in a list in a human readable format. So that's the same thing as running LS -l-dash H at your terminal.
So that's how we can call external programs from our now one thing that I really like to do for these sort of bioinformatics pipelines is it's a pain to go and install all those programs yourself, especially if they involve a lot of dependencies or if you have to actually build them from scratch, which is often the case with these sort of custom bioinformatics programs.
So Docker is really useful for dealing with that problem. So what is Docker well according to the documentation is to Standalone executable package of software that includes everything you need to run an application. So the upshot of that is if you use Docker in order to use a particularly a program, you don't actually have to install it or is dependencies. You can just run it from Docker, okay. So as an example, there's a program called Fast P which is often used by bioinformaticians for cleaning DNA sequences. And so this is the name of the image for to for fast p and the part after the colon here is what's known as the tag, which is essentially the version of that program.

So instead of installing fast P from scratch on a machine if we have Docker installed. All we have to do is give a Docker command Docker run the name of that image and then the name of the program that we want to run and it'll do it and it's tag at the specific version. So that's great for reproducibility. And I don't have time to discuss all the details of Docker today, but this is a recommended resource if you're merged if you're interested on on that.
Okay, so it's great that these Dockers containers exist. But where can we find them? So this slide is going to be mostly useful for Again by for people doing bioinformatics. I have some recommended resources the top the one I would definitely tell you to go to is bioconda. There's also one that's that sounds really similar called bio containers and at some point they ended up sharing pretty much all the same containers, so they're kind of equivalent. But if I am biochanda easier to use. Um and an important point about these sources at that. These containers are actively maintained and tested so that you can be pretty sure that you know, it will run correctly and and not cause any security problems. Okay, and that's which brings me to the other one.
I want to mention which is Dr. Hub. So Dr. Hub is kind of like GitHub in the sense that it's this massive repository for Docker containers or images and anybody can put one there. So it you can find lots of stuff on there but there's no guarantees about those images themselves and you should be aware that when you run Docker on your computer within that doctor container, it is running as a root. So if you give it access to things on your computer, it could potentially do nasty things to them. So you always have to be aware of. You know who developed this image and if it can be trusted so so do keep that in mind. nd of course, if you can't find a container you can make your own I have to do that all the time again, I don't have time to go into that today, but there's a link here in my slides with some more information. Okay, so we're going to be using Docker to run these programs and I want to do that from R. So, how can we do that? Well, there's a really nice package to help you do that called Babble whale and it has a function called run sounds a lot like process X run and it has similar syntax the process excellent. The only difference being that before the name of the command and the arguments we're going to provide it to container name and then it will run that command in that container using Docker on our machine. So for example, if we wanted to run the ls command we could with a really lightweight container just as demonstration. We could run that with that container called Alpine and so when you do this on your computer, it will download that Docker container and then run LS in the docker container. Now and that's just the toy example, but if you're to actually do this one wrinkle here is that you're not going to see the same thing as you would get from running LS at your terminal. So why is that like I thought we were doing this to be reproducible. Right? Well, that's because when you run LS and Alpine it is listing the contents that it sees in that Docker container and it turns out that the file system of the docker container is isolated from the file system of your machine. So whenever you want to actually work with files in Docker, you need to somehow connect those together and the way we do that is called mounting. Okay. So when you do that in Docker that's done with the dash V option. And so this is called volume mounting.
And so for the dash B option you then provide it with something a string that looks like this where on the left side that is the path to a local director or file on your computer that then you want to connect to a directory on the inside of the docker container on the right side of the colon, okay.

So basically you have to be aware of where things are on your own computer, but also the file system of Docker container, right? So for example, if we wanted to get LS to actually show us what's in our current working directory using Docker we could do something like this where we would get our current work during working directory and R with the gate WD save that to WD and then in that Babble whale run function there's a argument called volumes where you can provide the volume mounting. So here I have a construction where I'm pasting the current work during directory into the working directory and Docker and then it will show us that right.But this is kind of awkward and it gets really confusing. If you have a lot of files and folders that you're working with so it's kind of a pain to use. So what I actually did was I wrote another function called run auto mounts that can make that all go away. So I contributed this to the babble well package and it basically looks the same but the only difference is when you give it that Vector of arguments to the command, you can provide a named vector and if any of those names are file then that's going to tell baby. Well that hey this is a local file that needs to be mounted and it will do that for you automatically. So instead of that sort of nasty construction that we had on the last slide you can do something like this where you just give it a name to vector and you say file equals WD and also when I say file that applies equally to files or directories it to Docker. They're the same thing and then it'll mount it automatically and you don't have to worry about all of that.

Okay, and this is an example of some example code doing that with the fast P program that will clean up some some DNA sequences. Okay. Um now I would take that a step further and instead of just using run Auto Mount as a function in the target's workflow. I would put that inside its own custom function. So I'm calling that trim reads and it's taking some input files and then telling me the location to write the output right and so at the end we're going to get a nice what it in practice we would get something that looks like this which is a pretty clean function to trim reads. And again, it should work on any computer and you don't have to install fast yourself. And of course, what we're going to do at the end of the day is write a bunch of custom wrapper functions like this and our functions. R and then we'll have our targets.r and we'll you know set up the workflow and again using the tar file to track the contents of those files that get written output by each function.

Okay, so one final detail I need to cover is the environment for R. So Docker is great for maintaining reproducible Computing environments. And what I would normally do in a typical targets workflow is I would make with custom Docker image to run targets from but here we are calling Docker with from targets. Right? So like you can't call Docker from Docker or at least it's really frowned upon as a security practice. So what what to do my solution is to use conda and R, so I'll try to I'm running out of time so I won't have time to go into this in detail, but briefly.

So conda is a package management system that was originally developed for python. But now it can be used all sorts of programs. There's some more information on the links in this slide.

It uses an environment.yaml file to define a package versions. So here we are saying that we're using R and with that comes with are in at zero that one done at 0.15. Um, and then we can once we have that yaml file set up if conda is installed. We can create a Conde environment and then activate it and now we are in our That comes with R 0.15. Okay. So why are in well that's because RM is in the system that I use to maintain versions of our packages and that those are

saved in this RM Dot Lock file. And so that's where we can save our versions of targets, etc. And within our you use these functions to run RF, okay?
So I have a git repo that puts all of these pieces together to run a variant calling workflow, which is a yeah, bioinformatics style workflow that takes DNA as input and then tells us where the varying sites are across different DNA sequences, which is based on a data carpentry genomics lesson and in short you could run this and now entire analysis from your command line with these six commands, so we would clone the repo enter the repo create the Conde environment activated and then run targets and it should do everything for you with the exact same package versions and program versions that I've specified in the code.
And that is it. So I'll take any questions. I guess we're doing questions at the end, right? Okay.

Yani: Yes. Yes, you have several in the document if you want to look. Already answer some of them. So now it's Will turn. So when you want to start.

Will (00:41:09): Yeah, thank you so much, Yanina and Eric and Joel for for having me on and and sharing those wonderful presentations. You really fleshed out the use case and and that's that's really exciting. Can everyone see my screen? Great. So I'm Will Landau I created targets and a few packages in the targets ecosystem.
I am a Bayesian statistician by training and I became interested in pipeline tools for R when I was working on my dissertation in 2016 and had all sorts of issues trying to manually develop something like a pipeline to manage all these computationally intense Markov chain money Carlo runs and you know fast forward a few years. We have targets and Incredibly thankful for the for the the community that's it's developed around this and through through rOpenSci this is a really gratifying to seem to share.
I'd like to talk about challenges debugging in in targets. And the reason I this is a special topic of its own is because that our code is easiest to debug when it's interactive. When you have all your data objects and functions at your fingertips. You have a small bit of code that runs very quickly and you can experiment with an interrogate any any piece of it with targets things are at the opposite extreme. There are layers and layers of encapsulation on top of the actual work that gets done in a pipeline. So targets one of the first things it does is create an external process to start everything to protect the pipeline from any detritus in your local interactive environment that might accidentally corrupt some of the objects or hashes. There is the data management automatically saving and loading the data which which targets does on behalf of the user there is environment.
Judgment, so there's there's a just-in-time environment that each Target actually creates when it evaluates code and there's of course the high performance Computing that that Eric talked about and the the system libraries that Joel talked about and there's special air handling procedures that targets goes through and special error handling techniques that it does all this exists to more to make the pipeline more automated and more reproducible.
And usually that's a good thing when it works. It's a good thing it it grants trust in the analysis. It it affirms that your output is syncs with your Upstream code and data and you're the models and and code that you have are are directly connected at and with with the output that you have

Right Here and Now. Unfortunately, the very layers that make that Automation and reproducibility possible also make it difficult to debug their inherently at odds.
Fortunately, though there are a few things that you can do to make this debugging process easier. So here is just a few of the workarounds and built-in features of targets that that lets you do that. So there's an option that lets you. In spite of any errors that you may get finish the pipeline anyway, and if you're running a huge simulation study some of your models work and in fact, if most of your models work and only one or two fail, there's a way to plow through all that and still aggregate all the all the results of the end. They're of course error messages and warning messages that that Target stores there. There's there are techniques to to debug the functions that you write as part of the target's pipeline before you actually bring them into the pipeline their system issues. There. There's there there may be there are ways to to check the similar bits of code on on the packages and system libraries that targets depends on without actually fully running it with targets. So you can eliminate this pipeline framework from the possible sources of of errors and seems like if you're if you're in Joel situation with these bioinformatics pipelines that that might be that might be something you might be interested in running the the individual calls to process X or Babble whale or or even any of the other bioinformatics specific libraries in order to make sure that those to track down the air to something that's caused outside the target's Pipeline and then there are a couple of ways to to interactively debug the debug a Target as it's running as if the session we're interactive. There are ways that targets lets you do that and there's a feature that allows you to save a lightweight workspace file that that lets you recreate the session outside the pipeline with all the the right number generator seed and the packages and and functions in the environment that you're working with interactively.
So today, I'm going to focus on a few of these. I'm going to focus on finishing the pipeline anyway, and I'm going to focus on the browser and debug techniques.
The material for today, it's it there's more than I'll present today. There's their code examples in these slides that this GitHub repo and these interactive demo scripts go through line by line how to how to. Do these debugging techniques yourself? I'm going to start with the second one here and the demonstrate the browser like capabilities and I'm going to stop sharing this screen. I'm going to share my R Studio IDE let's get started with this example browser demo script, so first we're going to load the package and we're going to start with an example pipeline.
In this pipeline, we're we're simulating a longitudinal data set a repeated measures data set on on what for some experimental study possibly or even an observational study and we have an outcome and we have a we might have predictors or we have might have a factor that we're predicting on we're going to analyze it with General lightly squares and then we're going to return the we're going to return a result of a hypothesis test at the end and we're gonna put those functions together to with a function that simulates a data set and analyzes it once and returns very compact set of results because we're not going to just run this simulation workflow. Once we're going to run it several times at this tar mapped rep Target Factory. There is a question in the Google Doc about accumulation in in Target's pipelines. And I think that's referring to a map reduce pattern where you run a lot of steps in parallel that aggregate them all at the end and static branching and dynamic branching both have this capability and tar map rep is a way that combines the static branching and dynamic branching paradigms to allow for

simulation workflows where you might simulate multiple different static scenarios. So for example, what if we have 58 experimental units versus 70 experimental units and for each of those scenarios you run the simulation multiple times to get some aggregate of of results and it could be statistical power could be type 1 error. It could be could be other things. We want to look at in aggregate over a simulation. And that's what this workflow does.
Unfortunately, there's a bug in this workflow and it's stochastic. It's very unpredictable where this bug happens and it's it's really hard to track down because it doesn't happen very often. And and if you've run simulations studies of targets, you know what I'm talking about. We're we're on a thousand branches and maybe only one or two have this inexplicable error that rarely ever happens and you can't reproduce if you're just running your functions locally over and over again. Well this this demo script is for you in that case.
So let's run this pipeline as is right now. I'm gonna get more space here. It's setting up the main targets of the pipeline and it's running the pipeline and already we see an error.
And the main text here at the bottom is really is really what to focus on the the error message is all the way at the bottom complains about missing values in an object. I don't know what that means yet. Well I do but maybe maybe I maybe I don't want I'm running this for the first time and I think one of the first steps is to make this this enormous pipeline easier to debug with with that error and so well actually even before that there is an option that you can set to finish the pipeline anyway, and so, here's another version of the pipeline which says error equal to null and the options and this means that instead of airing out immediately the pipelines just going to return a null value and that gets aggregated together along with the successful results. And so you'll be able to just just get at the very end maybe not all the results you're looking for, but you'll still get an aggregate of useful useful output and you can even if it's not your final result that you trust you still get an approximation of the answer that you're looking for. So with that pipeline, let's begin.
And notice it's skipping the successful targets that already ran and some targets run into errors and the pipeline just keeps going these targets that return an error just return no and at the very end of it, which should be coming up soon. You'll be able to read in.
This fully aggregated analysis Target. And so when we read that notice, we only have 270 rows. You should see a lot more if this if everything were successful, but we might get enough to to at least at least. Get some insight about the about the the content of the research problem just to give an idea of where this analysis Target fits in if we look at the graph this is this is aggregate this analysis. Target is is an aggregate of these two Dynamic branching targets that are scenario specific.
We we note that the air is occurred in both of them. The the next step is to try to make this pipeline smaller and easier and in a in a batch replication workflow. It'd be good to just set one rep per batch. And also let's just pick one scenario to run in order to make this debugging easier. So we're gonna create a new version of this of this pipeline where at the bottom. We just take one of these scenarios and we set reps equal to one. So each Target is only going to run one simulation. And let's run it locally. In let's run this these Target sequentially until we get an error this may take a while you can you can also run it. On on a cluster in parallel if you like to get to this point, but now we have the name of a Target that aired out in our one rep per batch version of the pipeline and it and it good. It has this this error missing values and object. We

don't know what that means yet, but we're going to work with that. The next step in interactive if we want to interactively debug that while the pipeline is running we can set this debug option to be the name of that Target and then set a queue to make sure that it in spite of any other changes we might have made to the workflow that well we skip those targets anyway, and this is all this is always temporary. And so with that looks like is just these couple of options in tar options set.
We go to run the pipeline first. We're gonna restart the session just to get rid of any to try this in the environment that may the May under these conditions corrupt any Upstream targets that might have run successfully. Because now we're going to run the pipeline with call our function equal to null this can be run with tar make tar make cluster and Q or tar make future and what this does is prevent the encapsulation behind this call our process usually not recommended. But in this case if you restart your session, you'll probably be fine.
Let's run the pipeline and what this debug option is going to do is no matter whether this target whether the other targets are running on jobs in the cluster this particular Target that you chose to debug when you run it. It's going to send you directly into an interactive debugger and what this is, you know, you're in a debugger when you have this browse statement here. This is the this is in the middle of an executing pipeline there. There's a lot of great stuff written about interactive debugging in the advanced our book.
And I highly recommend it there's a link to it here. But for now we're going to assume that knowledge and we're going to debug the the function that were that were that aired out because we're going to try to find the source of the air.
So we're gonna say there are a bunch of ways you could do this. You could run the command of the Target in this case. It's wrapped around some Target types function. We don't really know what this means. It's more convenient in this situation to to pick one of the functions that you wrote to to debug and then proceed along until the next breakpoint and already we noticed that these missing values were were produced and if we try to run this model with GLS, we got the the same the same error missing values in object.
Now here we might have we might have some more clues because this na dot fail dot default appeared where it didn't before. So maybe that was from the GLS checking for missing values in the outcome and sure enough we can we can verify that because we're in this interactive session with objects in the environment. We can see what actual data that we had and we can check things like
And sure enough they're missing values here and we can confirm that that's the source of the problem by filtering them out so we can actually check interactively and verify that we've we've fixed the bug when we've identified the correct solution, so I’m going to filter. Let's see.
And now if I run this command here we get maybe not the correct answer, but at least we don't get an error. And so this pattern really helps no matter how large your pipeline gets. This is a really useful tool for for debugging those elusive kinds of errors.
And like I said, there are a couple more of demo scripts like this in this code repository and there's there's more information in the in the the debugging chapter in the user manual. That's why that's why with every error message in targets at the pipeline level. There's this link to this guide it goes through a lot of these techniques. There's also a chapter that I recently wrote at the beginning this year on on, you know, helping me help you asking for asking explaining some

some helpful tips as far as how to how to ask for help the more empirical the process is the quicker we can get to a solution together.
And that's all I have.

Yanina: Thank you. Thank you so much. We we are on the top of the hour, but we have several questions on the doc. Thank you so much to the others people on the call that are already answered some of the questions. So if we if you have a couple more minutes we can do some of this question if you are okay with that.

Will: have some more minutes and I'd be happy to answer the other questions within the next day or two the ones that didn't get.

Yani: actually we always let this share doc open so people can continue writing and giving your answers. So I just pick one question that two people do similar similar questions. Nicolas and Elio are asking about what are the best practices for using Targets in large data scenario. Elio ask for tricks and tips to things to consider when working with relatively large data to optimize RAM usage.

Will: Right, those are great questions. And they and they do come up a lot and with large computation. There's very often large data too. And one of the first places for maybe medium-sized data that I would check are the the storage formats that are available in targets. So the tartar Target function the tar option set function. They have formats that allow you to customize how Target object is stored.
The default is to try to save the output object as as an RDS file that may or may not be what you want. If the data is the data sets are really large you can choose things like feather and parquet formats for large data frames FST as well. You can choose the Qs or quick cerealization format for large non data frame objects. There. There are a few others and there are ways to also write your own actually. There's you you'd have to specify a function to save to read and then to Marshall on Marshall the object and if the data is larger than even that I would recommend Creating file targets with with formide equal to file or as Joel said that the tar file Target Factory in the Target types package. And so and then and then Downstream targets can if the file is support something like this you'd be able to read only that for example the rows of the data that you need this this actually relates to the then The Logical question is is well should I just have my data in a database and there's a question in the Google docs about that as well. And that's for if the data database exists prior to the pipeline is an input to the pipeline. Then you can have a Target that that makes a query to the database and then uses a hash to if you want to get really Advanced with it you can you can write what's called a cue with the tar Q function the depends on the hash of the data table that you're reading from and that can help targets decide whether that Target is outdated or up-to-date.
And for the most Downstream targets it a it's natural to to maybe maybe save an output to a different table and a database it is tricky though in some cases you want to really have be

careful that you avoid circular workflows, you know writing to the same table that you read from because targets does assume that each Target is an immutable step.

Yani: Okay, thank you. Thank you. Will we have a question from Nathan: Are these sequential Target branching patterns? something like accumulate? There would still be a benefit of steps that have already successfully run. As an example might be a random walk where each steps is captured as a Target. I'm asking about sequential not parallel branching. Where the next branch operates on the result from the last branch run.

Will: I see. Okay. That's that's a bit clearer. Now. I thought accumulate meant something like mapreduce which is highly parallel there don't really exist Target factories to make that convenient currently in in the in Target types the way the same way that tarmap rep does but I think that that is a pattern that that comes up every so often.
I'm not aware of something that that makes that abstracts that but I think certainlyit's and it's possible to write your own Target Factory that would simplify that down. actually in some in some cases there's there's. Yeah, I'll have to think more about that. But there there I think Noman I've talked about a Atari download Target Factory where where the first step is to download a file. The next step is to load it into the targets data store there there are little patterns like that. But I think the more massive ones or the parallel Target factories like tar map rep.

Yani: Thank you. Nathan say thanks on the chat. Noam asked what are the current state and near-term expectation for semi transient Cloud workers for targets.

Will: I'm so glad you brought that up. That's been my on my mind for years. I've I was talking with Kira Mueller at one of the our studio conferences back in in 2018 or 2019 and We've been I mean so far that the ecosystem makes it really easy to build on fully persistent or fully transient workers.
In between is a bit harder Rich Fitz John who created the the Remake package which is a predecessor to targets. He he's got this package called rrq that does have these semi-transient workers based on redis and I've been I've been slowly working on this package called crew that I I want to build on top of roq and make it easier to work with different kinds of workers what that is. One of my goals for this year is to is to make significant progress on that. It is slow going because somebody other things demand my time, but I really care about solving this problem. I think that the way that it's going you know, I've got I've got some I've started writing code to rap around read a server instances and I'm I think that this problem because I'm more familiar with with traditional clusters. Then Cloud compute. I think it's I think this problem is going to be solve sooner for HPC systems traditional HPC systems than it is going to be for Cloud Computing Services, like batch and Lambda and ec2s and kubernetes. It's a the cloud compute is a huge learning curve and like Eric said it's it's really it can get expensive too. And so that's that's gonna be that's gonna be a mountain to climb.

Yani: Thanks. and thanks Eric and Joel for answer the questions on the Google Doc. Okay, I go into do one more question and then we are going to to close the call. We already keep you several more minutes. Sergio Olmos Is it possible to use something like config R package to provide different values say size of the data a number of models inside the target pipeline when the pipeline runs in your local machine and when the pipeline runs in an HPC?

Will: And that's a great question. And I mean, especially a lot of those. Those the things that you might set with that you mentioned with config adapting to different Computing platforms. I think that's a that's a great way to organize it. If if those if if those settings control things that are set outside of a pipeline or even in tar options set, then it could be it could be a nice way to use something like like the config package where you have this yaml file of environment variables that you insert in various places, so It would be trickier to set things that the workers need to pick up because they're gonna run they might run a different computers within high performance Computing type scenario or at least on an HPC system. It is easier to control. Let's say arguments to Tar make and by the way targets does have its own projects projects multi-project system that's built on its own set of of environment variables in its own yaml config file and that that was inspired by the config package. It's it doesn't it. It's not implemented exactly like config but it borrows a lot of the elements from it a lot of the a lot of the behaviors and it's a way to set the default arguments to Tar make and other functions the project chapter and the user manual describes how to use it and that the functions are tarconfig set and tarconfig get.

Yani: Okay. As I say we are going to close the call.

Yanina's closing words:

Thank you so much Will, Joel and Eric for presenting today, and thank you all for joining us. I'll leave the doc open for edits for the next day so please feel free to add your responses or notes, and the speakers will add their responses as well. The video of this call and any other resources will be posted at ropensci.org/commcalls and we'll post from rOpenSci mastodon when they're up.

If you're looking for a friendly and productive way to kickstart 2023, join us for rOpenSci social coworking and office hours on Tuesday, February 07 at 9 AM PST.

Thank you again for your time and attention. On behalf of rOpenSci, I wish you a good day or night, wherever you are in the world.