Knowledge Progression handling RNA-seq datasets

Aims

I want to see if I can replicate results published on the basis of a bulk RNA-seq dataset in terms of:
- differential gene expression (DGE) analysis
- GO term analysis on differentially expressed genes
- Functional enrichment if possible
- enrichment in disease-related genes or SNPs
- gene network analysis

Finding a suitable study

I searched for clinically relevant, loosely cerebral cortex development-related RNA-seq datasets with attached publication. I selected only datasets uploaded last year and whose publications don't do highly in-depth analyses, as I believe that me working on this data will bring the biggest contribution to expanding our knowledge horizon. As to why I selected datasets that were associated with a publication: this was for me to be able to cross-check the results of my analyses and make sure I am performing my analysis properly.

I found this study: GSE167193 on GEO at NCBI. This study analyzed a potential connection between a pregnancy disorder known as pre-eclampsia and autism spectrum disorders. Pre-eclampsia is a common and dangerous increase in maternal blood pressure that can occur after the 20th week of pregnancy. One in 20 pregnant people are affected by pre-eclampsia, which has a severe negative impact on both the pregnant person and the fetus. There is no treatment known to date, except for the delivery of the placenta, and very little is known about the long-term consequences on the development of the child's brain.

The authors used a pre-eclampsia mouse model in which the disorder was induced in pregnant mice by treating them with a compound called L-NAME. The mouse offspring from mothers that were exposed to L-NAME behaved in ways reminiscent of ASD and scored accordingly on several behavioral tests that are thought to assess metrics related to the condition.

To better understand the relationship between the two conditions, the authors sequenced RNAs from the cortices of mouse embryos at day E 17.5, two days before birth, from mother animals that had or did not have elevated blood pressure, and similarly from the hippocampi of adult offspring. The authors generated cDNA libraries from the cortical RNAs of control and experimental condition embryonic cortices using a kit for stranded mRNA detection. This means that the kit enriches for poly-A-tailed mRNAs and also captures information regarding which genomic DNA strand the mRNAs were transcribed from. This is an excellent fit for my purposes, as I am, for the time being, interested in coding transcripts. Then, the cDNA libraries were fragmented, and the fragments were sequenced on Illumina Hiseq X machines, a process that generated millions of paired-end 150 bp reads (21 million, to be a little more precise [link to post where I open the FastQC files]).

I focused my investigation on the embryonic cortex data, as this is a system I am more familiar with from my own research work.

```
iweber@Lenovo-Ioana:~/Datasets/Pre_eclampsia_mice$ vdb-validate SRR13761520
2024-03-11T18:10:23 vdb-validate.3.1.0 info: Database 'SRR13761520' metadata: md5 ok
2024-03-11T18:10:23 vdb-validate.3.1.0 info: Table 'SEQUENCE' metadata: md5 ok
2024-03-11T18:10:23 vdb-validate.3.1.0 info: Column 'ALTREAD': checksums ok
2024-03-11T18:10:24 vdb-validate.3.1.0 info: Column 'QUALITY': checksums ok
2024-03-11T18:10:25 vdb-validate.3.1.0 info: Column 'READ': checksums ok
2024-03-11T18:10:25 vdb-validate.3.1.0 info: Database 'SRR13761520' contains only unaligned reads
2024-03-11T18:10:25 vdb-validate.3.1.0 info: Database 'SRR13761520' is consistent
iweber@Lenovo-Ioana:~/Datasets/Pre_eclampsia_mice$
```

I downloaded the datasets directly from the entry of this study on GEO in SRA format. As I found out, SRA is an NCBI-specific way to compress FastQ files. These are the ones with the raw reads from the sequencers and I learned that these are precisely what I will need for performing differential sequence analysis. The processing of the files to any other format, such as SAM or BAM, or anything else except for basic quality control, may introduce biases that affect subsequent data analyses.

In the Windows Subsystem for Linux (WSL), getting the SRA Toolkit and SRR run files from the preeclampsia study

Using the documentation at NCBI, I found that one needs a suite of programs called SRA Toolkit in order to losslessly convert the SRA files into FastQ files. I installed the toolkit using the instructions on the GitHub page of the NCBI, https://github.com/ncbi/sra-tools/wiki/ . I decided to do so under Ubuntu running on the Windows Subsystem for Linux (WSL). I have read that, for optimal memory usage, it is preferable to create a virtual machine running a Linux distribution rather than the WSL. However, given that this was my first shot at analyzing a relatively small SRA dataset (~2.8 GB), I concluded that this is good enough for starters.

I downloaded the SRA toolkit as a .tar.gz archive for Ubuntu from one of the FTP servers of the NCBI:

`wget --output-document sratoolkit.tar.gz https://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/current/sratoolkit.current-ubuntu64.tar.gz`

then extracted it in one of my Ubuntu folders with the tar utility (included in Ubuntu),
`tar -vxzf sratoolkit.tar.gz`

and could afterwards access its tools over the command line.

As per the instructions of the GitHub page, I modified the PATH variable in my system. This essentially tells Ubuntu that there is a shortcut to the folders where SRA Toolkit has its binaries (bin directory) so that I can then easily access the individual tools and commands from the toolkit via the command line without needing to constantly change directories to that in which I extracted the SRA toolkit.

```
`export PATH=$PATH:$PWD/sratoolkit.3.1.0-ubuntu64/bin`
```

This should tell the shell that, when looking for binaries (tools, programs, etc that are executable), it should also look in the respective folder of the SRA Toolkit.

To verify whether the binaries can be found by the shell, I used

```
fastq-dump --stdout -X 2 SRR390728
```

...which worked as expected with returning the path to the directory of binaries precisely until I restarted the terminal :) I then found out that "export" only does a temporary modification of the PATH environment variable, which isn't kept in the global namespace. So, after advice from Christoph, I found out I need to only add the exact same line, minus "export", into the .profiles file, which should save any customized variants of the PATH variable. And now I have the SRA Toolkit set up to work from any directory, and even after restarting the shell.

Additionally, I created a handy alias for changing the directory to where my datasets are stored by adding the following under "Some more aliases" in the .bashrc file:

`alias mydatasets='cd /home/iweber/Datasets'`

So now I can go straight to that folder with only typing "myd" and hitting tab.

I also undertook the Toolkit Configuration as directed by the GitHub page of SRA toolkit using `vdb-dump -i` .double check command One noteworthy difference to how the configuration window looks like for me vs the website is that I do not have, under Main, the frame with options regarding to Workspace name and Workspace location.
I wonder if this is a matter of working on WSL instead of a full Linux distro?

However, in the tab Cache, I did set the location of the user-repository and process-local to be my Datasets folder, so that I may always find my files easily. I also created a subdirectory for my current study so that my data stays organized.

WSL - Extracting FastQ files from the SRR archives

And now that the tools are set up, I went on to try and extract the FastQ files from my dataset in SRA format. In my Datasets directory, I used the fasterq-dump command together with my dataset as a command-line argument:
`fasterq-dump SRR13761520`

And waited. I was afraid nothing is happening and I just went into some infinite loop, but then I navigated in Windows to the Datasets folder in WSL and saw that a temporary directory had been created. Since the terminal also didn't show me a fresh command prompt, I assumed it was just working intensely in the background to convert the data. Also, my ventilators were going crazy. s, when I checked the Task Manager, I saw that nearly 60% of my memory was in use by a process called "Vmmem". Sure enough, a brief web search uncovered that this happens precisely when using WSL, and is the Windows process responsible for the virtual memory managementDouble check!. All of this was happening before switching my RAM chips from a total of 32 GB to a total of 128 GB :) Yes, I am aware that I will need to set up a few more things, such as hyper-threading, for the new RAM to work as effectively as possible, but I was reluctant to do this at first, as my first experience after getting my laptop had been it crashing and refusing to start Windows upon installing a VirtualBox virtual machine and tinkering with the virtualization settings).

I had read on the GitHub page of the SRA Toolkit that the extraction of FastQ files requires a RAM size up to as much as 10 times the size of the original SRA file, at least during the extraction process (scratch space - had run into issues with this with Photoshop before, which kept complaining that the scratch disks were full and it therefore couldn't save whatever new artsy monster file I had created). So I was expecting Task Manager to tell me that around 25 of my 32 GB of RAM (~78% of it) were in use by the Vmmem process. The process started out at around 50%. Half an hour of observation later, I realized that it kept increasing over time. I did see it go up to a max of 75.6% with a fairly steady increase over time, so I assume the Toolkit kept adding info to the scratch disks as the process progressed. Looking in the Resource Monitor of Windows, vmmem said it was using a Commit of 10,470,000 and a Working set of about the same size. I suspect Task Manager is not as accurate as the Resource Monitor ?

To check the progress of the unpacking, I peeked at the size of the temporary folder in the SRA folder of this study in Datasets. When I first looked at it, some 3-4 minutes after starting the extraction tool, it was at around 3.8 GB. Some 10 minutes after starting the extraction, it was at 6.5 GB.

At this point, I started wondering where exactly WSL was storing its files in a physical sense (I know, I know, should've done so sooner, etc). It being a subsystem of Windows, I strongly suspected it would also be on my Windows drive, which, unfortunately, is the smaller of my two SSDs, at 250 GB. And indeed, when checking the properties of my Windows drive, I realized that the free space on this drive was decreasing. Being fairly close to the

size limit of the drive (~10 GB free), I was concerned that this might completely kill Windows if working on such a full drive or result in a corrupted, incomplete extraction of the FastQ files. So I started uninstalling any programs I saw were using a lot of space with their files and freed up some 10 more GB of RAM.

I also noticed that the Toolkit had also created another folder under Datasets called "sra", with a .sra.cache file of a only slightly larger size as the original SRA dataset (2,829,033 KB for the dataset and 2,829,054 KB for the .sra.cache file). This did not change in time.

It took around one hour for the extraction to finish, and I then got this:

```
iweber@Lenovo-Ioana:~/Datasets/Pre_eclampsia_mice$ fasterq-dump SRR13761520
spots read      : 21,192,945
reads read      : 42,385,890
reads written   : 42,385,890
iweber@Lenovo-Ioana:~/Datasets/Pre_eclampsia_mice$
```

And bam! I had two FastQ files in my study directory, each at exactly 7,954,345 Kb, so 7.9 GB and a total of around 16 GB. All in all, that's around 5.7x larger than the original SRA file.

But why were there two files with the same accession number but with _1_ or **2** appended to the file name? The answer was fairly easy to find: these were reads from paired-end sequencing runs, as indicated on the NCBI website for this dataset and the paper's methods. Sequencing both ends of the cDNA fragments resulting from the cortex RNAs allows for more precise alignment [source] and is thus more useful especially in detecting alternative splicing isoforms, which may be of interest to me later.

I opened the two behemoth files by ~~torturing~~ convincing Notepad++ to do so, and, at first glance, they looked identical.

After this step, I also realized I had severely underestimated exactly how many files I'd have to download and extract, and to what space requirements that would amount to :teary-smile:

So, the first thing I did was to...get a larger SSD for my OS. I initially had a 256 GB NVMe and switched to a 4 TB instead.

Temporary fix for space issues: change location where WSL stores its files

But, for the time being: how could I change where Windows saves the files of the WSL? Thank goodness that I'm not yet at a level at which my questions haven't yet been asked by anybody on the Internet. I found this set of instructions by NotTheDr01ds on SuperUser: https://superuser.com/questions/1736576/change-the-storage-location-of-a-wsl2 I decided to execute them on the basis that they also create a backup image of Ubuntu on the WSL as I had it set up before, that the code was clearly commented and easy to understand in terms of what environment variables were being modified how, and that the user seemed to have taken extra time to spell out precautions and build in safety checks. This is what I ran:

```
PS C:\Users\Ioana> wsl -l -v
  NAME      STATE           VERSION
* Ubuntu    Stopped         2
PS C:\Users\Ioana> $WSL_ROOT="D:\WSL"
PS C:\Users\Ioana> $WSL_IMAGE_PATH="${WSL_ROOT}\images"
PS C:\Users\Ioana> $WSL_OLD_DISTRO_NAME="Ubuntu"
PS C:\Users\Ioana> Get-ChildItem HKCU:\Software\Microsoft\Windows\CurrentVersion\Lxss\ |
>>     ForEach-Object {
>>         (Get-ItemProperty $_.PSPATH) | Select-Object DistributionName,BasePath
>>     }

DistributionName BasePath
---------------- --------
Ubuntu           C:\Users\Ioana\AppData\Local\Packages\CanonicalGroupLimited.Ubuntu_79rhkp1fndgsc\LocalState


PS C:\Users\Ioana> $WSL_NEW_DISTRO_NAME="Ubuntu_WSL2"
PS C:\Users\Ioana> $WSL_INSTANCE_PATH="${WSL_ROOT}\instances\${WSL_NEW_DISTRO_NAME}"
PS C:\Users\Ioana> mkdir $WSL_IMAGE_PATH


    Directory: D:\WSL


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         3/14/2024   9:35 PM                images


PS C:\Users\Ioana> mkdir $WSL_INSTANCE_PATH


    Directory: D:\WSL\instances


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         3/14/2024   9:35 PM                Ubuntu_WSL2


PS C:\Users\Ioana> wsl --shutdown
PS C:\Users\Ioana> wsl --export $WSL_OLD_DISTRO_NAME "${WSL_IMAGE_PATH}\${WSL_OLD_DISTRO_NAME}.backup.tar"
Export in progress, this may take a few minutes.
The operation completed successfully.
PS C:\Users\Ioana> wsl --import $WSL_NEW_DISTRO_NAME $WSL_INSTANCE_PATH "${WSL_IMAGE_PATH}\${WSL_OLD_DISTRO_NAME}.backup.tar" --version 2
Import in progress, this may take a few minutes.
The operation completed successfully.
PS C:\Users\Ioana> wsl ~ -d $WSL_NEW_DISTRO_NAME
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.146.1-microsoft-standard-WSL2 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

This message is shown once a day. To disable it please create the
/root/.hushlogin file.
root@Lenovo-Ioana:~# sudo -e /etc/wsl.conf
root@Lenovo-Ioana:~# wsl --shutdown
Command 'wsl' not found, but can be installed with:
apt install wsl
root@Lenovo-Ioana:~# exit
logout
PS C:\Users\Ioana> wsl --terminate $WSL_NEW_DISTRO_NAME
The operation completed successfully.
PS C:\Users\Ioana> wsl --set-default $WSL_NEW_DISTRO_NAME
The operation completed successfully.
```

And it seems to have worked. Also, this was my first time seeing both Windows ***and*** Linux command lines mixed in the same window and, as the beginner that I am, it blew my mind. I realized that, basically, all I'm doing when starting the Ubuntu terminal I can also do from PowerShell window, provided that I started the WSL within it.

With WSL open in the PowerShell terminal, I did several checks as to whether the new distro at the new location was working properly:

```
PS C:\Users\Ioana> wsl ~
iweber@Lenovo-Ioana:~$ wsl.exe -l -v
  NAME            STATE           VERSION
* Ubuntu_WSL2     Running         2
  Ubuntu          Stopped         2
iweber@Lenovo-Ioana:~$ myfiles_exercises
iweber@Lenovo-Ioana:~/ABI/Exercises_Ioana$ ls -lat
total 48
drwxr-xr-x  4 iweber iweber 4096 Mar  4 10:47 Linux
drwxr-xr-x 12 iweber iweber 4096 Mar  4 09:25 .
drwxr-xr-x  2 iweber iweber 4096 Mar  1 12:32 Bash_scripting_Ioana
drwxr-xr-x  3 iweber iweber 4096 Feb  6 09:56 Databases
drwxr-xr-x  2 iweber iweber 4096 Feb  6 09:55 Biopython
drwxr-xr-x  7 iweber iweber 4096 Feb  6 09:46 ..
drwxr-xr-x  4 iweber iweber 4096 Feb  6 09:45 Alignments
drwxr-xr-x  2 iweber iweber 4096 Feb  6 09:45 Analysis_and_Stats
drwxr-xr-x  2 iweber iweber 4096 Feb  6 09:45 NGS
drwxr-xr-x  6 iweber iweber 4096 Feb  6 09:45 Python
drwxr-xr-x  2 iweber iweber 4096 Feb  5 13:01 Structural_Bioinfo
drwxr-xr-x  2 iweber iweber 4096 Feb  5 13:01 Working_w_Strings
iweber@Lenovo-Ioana:~/ABI/Exercises_Ioana$
```

I started WSL, then tested which distro was running using the code provided by NotTheDr01ds. In addition to that, I checked whether I could still use an alias I had set in my .bashrc some time ago, which simply changes the working directory directly to one where I store files for the ABI course. I also listed the contents of the directory, just to be extra sure that they are what I know them to be and that my default user is still the owner and has all of the permissions, as I know this can be difficult to alter should I not be the owner any longer ***source?***

And I then de-registered my old distro from WSL.

A consequence that I hadn't anticipated is that now I cannot use the Ubuntu app from the store directly any longer, at least not without creating a completely new distro...but that's easily fixed by running Ubuntu from the Windows Terminal instead, which I've taken to doing.

Result: successfully changed location, freeing some space on my currently plighted OS drive. I will need to change it back once I exchange the OS drive to a larger one, but at least the datasets I have stored in WSL and all of my course materials are safe for now on the other SSD.

Considering to make regular .tar/VHD backups of the WSL on the second SSD just to be on the safe side

I then wanted to get and extract all of the other SRA archives related to the E 17.5 cortices in this study. To do so, I wrote a Bash script containing the commands for fetching the remaining SRA archives (`prefetch` command of the SRA toolkit) and to unpack them (`fasterq-dump` command), set it to executable, and started it from the shell.

I checked the progress on occasion via TeamViewer, and saw things moving along through the continual increase in the size of the export folder. As a side note, here I also saw that the WSL virtual machine was still using up almost half of my RAM during this operation, and I think this is due to the limitations that WSL has in terms of memory usage, at least when compared to a virtual machine, so I decided I must take the plunge and create one. (see linked post about VM)

Real solution: Making a VirtualBox virtual machine to run Ubuntu

Before taking the plunge to make a virtual machine, I made some very thorough backups of my system. I made a restore point, then also used File History and Backup and Restore to make images of my WinOS. Finally, I made a recovery USB in case I wouldn't be able to start the OS after making changes to the BIOS to enable virtualization. Maybe I wouldn't be this paranoid if tinkering with the virtualization options hadn't BSOD-ed my back-then three days old laptop right after setting it up out of the box for the very first time...so better safe than sorry.

Got an Ubuntu 22.04 image from the Canonical website, and installed it as VM with the name Ubuntu 22x64 using the Oracle VirtualBox. I allotted it 96 GB RAM, and 6 CPUs, to make sure it has all of the resources it might need to analyze large data sets.

I thought this worked at first, and then the issues started popping up like mushrooms after the rain: the GNOME terminal that comes with the distro didn't work, I couldn't update the Snap Store even though the update was recommended. I installed another terminal emulator called Guake and attempted to install the bcbio-nextgen tool bundle...and it got stuck with conda not being able to solve the environment (this issue seems to be known and caused due to R package dependencies that conda can't solve, or at least not quickly, see link). After stopping that process, I tried updating Python, which almost completely broke the entirety of the Ubuntu installation: the VM stopped showing me the Ubuntu desktop at all and only did so for the TTY terminal, with which I was not able to refresh the installation of the GUI desktop.

(Side note: I also tried the terminal emulator Terminator, whose options I also enjoyed).

I removed the VM completely and reinstalled everything from scratch...only to find out that, suddenly, my user, created during the VM creation process, had no `sudo` privileges and I had no way of accessing the root to give it said privileges. Removed the entire VM again.

Reinstalled Ubuntu 22.04 on a VM with the name ubuntu22x64. Gave it my username iweber and my usual password for the purpose, and selected guest additions from the suggested iso. Let's see if I now have sudo rights for my user... Nope.

Found a forum entry here saying that the issue can be that this unattended installation option that the VM provides creates an user without `sudo` rights. Skipped this option during the creation of the new VM, leaving me to install Ubuntu manually after the creation of the VM.

Did so, selected the keyboard and then to erase disk and install Ubuntu.

After the installation and restart, noticed that the first DOS-like screen had some errors talking about a graphics device and issues with unsupported hyper-visor, but Ubuntu started nonetheless and I could log in with user loWe (my user, with actual username iweber). ***LE: this keeps happening when the virtual machine powers up, but it still boots, and I am currently not seeing any other issues.***

And magic! I could even open the GNOME Terminal now!

I then let Software Updater update everything it said needed updating. It ran and I noticed it again giving some errors regarding Snaps...but then it finalized the update and said it needed to restart to apply all of the updates, so I let it do so.

After the restart, the terminal still works. I could also `whoami` myself AND `sudo whoami`, which, after providing my password, also let me set the user to root, so I now know I can install any software I may be needing down the line.

I went to the Anaconda website to find out how to download packages that I previously found out need to be installed for the Anaconda Navigator GUI (anaconda3) to work. The command I used in the terminal was:

```
sudo apt-get install libgl1-mesa-glx libegl1-mesa libxrandr2 libxrandr2 libxss1 libxcursor1 libxcomposite1 libasound2 libxi6 libxtst6
```

and it executed without me seeing any errors.

Tried to download the Anaconda Navigator installer Anaconda3-2024.02-1-Linux-x86_64.sh as recommended by the Anaconda Navigator page by using

```
curl -O https://repo.anaconda.com/archive/Anaconda3-2024.02-1-Linux-x86_64.sh
```

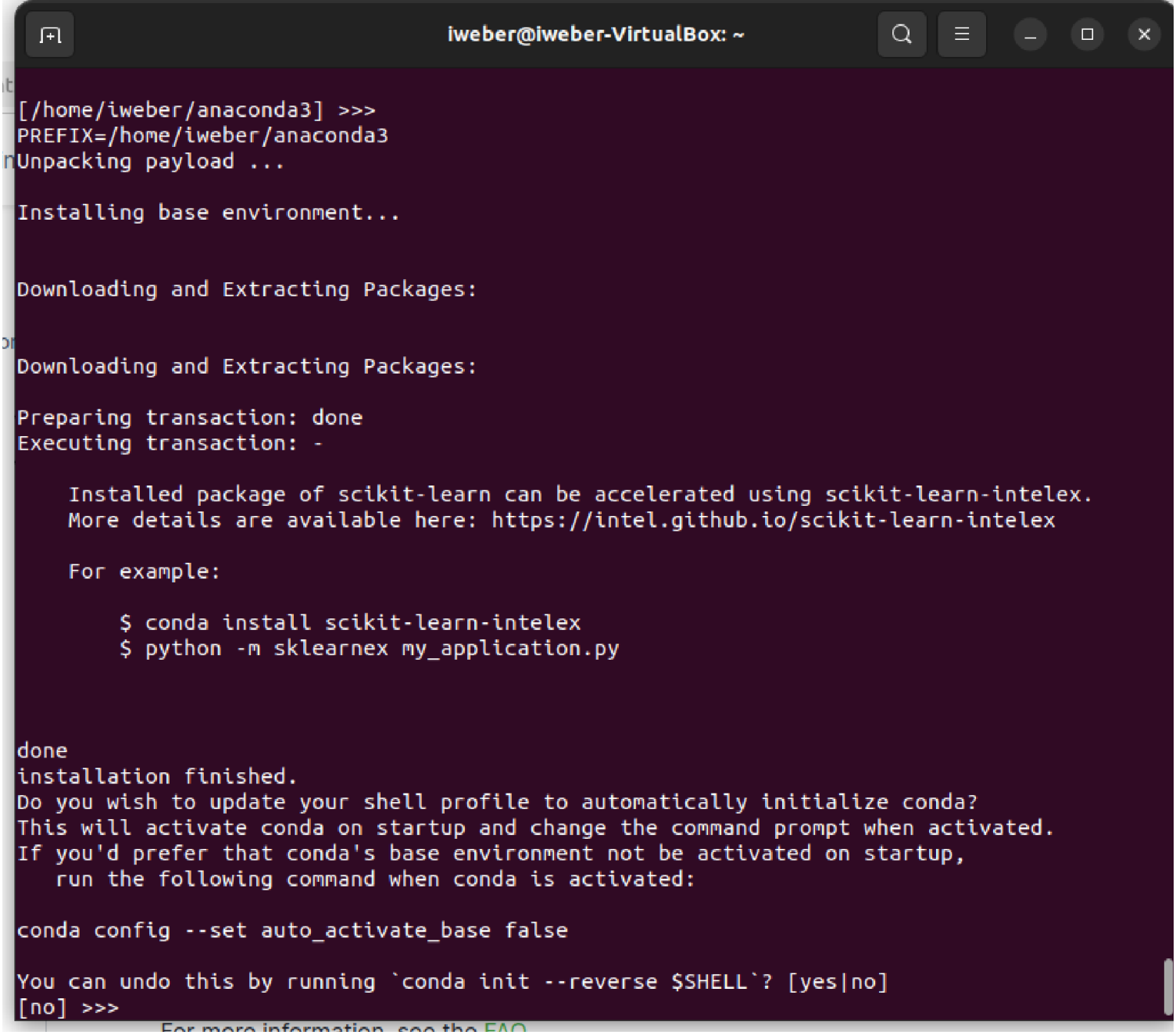
...and found out I first had to install the curl package (*cry-laugh*), which I quickly did with

```
sudo apt install curl
```

I could then finally run the curl command above to get the Anaconda Navigator. What was displayed during the installation was this:

```
iweber@iweber-VirtualBox:~$ curl -O https://repo.anaconda.com/archive/Anaconda3-2024.02-1
-Linux-x86_64.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  8  997M    8 82.9M    0     0  6819k      0  0:02:29  0:00:12  0:02:17 6633k
```

The percentage reached 100 and I had a new command line available, so I knew it was installed. I checked, as per the rec of the Anaconda Navigator install page, whether the checksum is correct using `shasum -a 256 Anaconda3-2024.02-1-Linux-x86_64.sh` (curl always downloads in the working directory). This check returned the expected SHA-256 hash for this version, so I knew I was good to go actually using the installer. Used `bash Anaconda3-2024.02-1-Linux-x86_64.sh` to run the installer, and what I saw was:


iweber@iweber-VirtualBox: ~
[/home/iweber/anaconda3] >>>
PREFIX=/home/iweber/anaconda3
Unpacking payload ...
Installing base environment...
Downloading and Extracting Packages:
Downloading and Extracting Packages:
Preparing transaction: done
Executing transaction: -
Installed package of scikit-learn can be accelerated using scikit-learn-intelex.
More details are available here: https://intel.github.io/scikit-learn-intelex
For example:
$ conda install scikit-learn-intelex
$ python -m sklearnex my_application.py
done
installation finished.
Do you wish to update your shell profile to automatically initialize conda?
This will activate conda on startup and change the command prompt when activated.
If you'd prefer that conda's base environment not be activated on startup,
run the following command when conda is activated:
conda config --set auto_activate_base false
You can undo this by running `conda init --reverse $SHELL`? [yes|no]
[no] >>>
For more information, see the FAQ

I confirmed that I wanted to let conda configure my shell profile so that it can be automatically initialized by entering `yes`.

```
done
installation finished.
Do you wish to update your shell profile to automatically initialize conda?
This will activate conda on startup and change the command prompt when activated.
If you'd prefer that conda's base environment not be activated on startup,
   run the following command when conda is activated:

conda config --set auto_activate_base false

You can undo this by running `conda init --reverse $SHELL`? [yes|no]
[no] >>> yes
no change     /home/iweber/anaconda3/condabin/conda
no change     /home/iweber/anaconda3/bin/conda
no change     /home/iweber/anaconda3/bin/conda-env
no change     /home/iweber/anaconda3/bin/activate
no change     /home/iweber/anaconda3/bin/deactivate
no change     /home/iweber/anaconda3/etc/profile.d/conda.sh
no change     /home/iweber/anaconda3/etc/fish/conf.d/conda.fish
no change     /home/iweber/anaconda3/shell/condabin/Conda.psm1
no change     /home/iweber/anaconda3/shell/condabin/conda-hook.ps1
no change     /home/iweber/anaconda3/lib/python3.11/site-packages/xontrib/conda.xsh
no change     /home/iweber/anaconda3/etc/profile.d/conda.csh
modified      /home/iweber/.bashrc

==> For changes to take effect, close and re-open your current shell. <==

Thank you for installing Anaconda3!
iweber@iweber-VirtualBox:~$
```

This seems to have worked. I closed and reopened my shell and checked if I can use the `conda` command to open the manager.

Success!

```
(base) iweber@iweber-VirtualBox:~$ conda
usage: conda [-h] [-v] [--no-plugins] [-V] COMMAND ...

conda is a tool for managing and deploying applications, environments and packages.

options:
  -h, --help            Show this help message and exit.
  -v, --verbose         Can be used multiple times. Once for detailed output,
                        twice for INFO logging, thrice for DEBUG logging, four
                        times for TRACE logging.
  --no-plugins          Disable all plugins that are not built into conda.
  -V, --version         Show the conda version number and exit.

commands:
  The following built-in and plugins subcommands are available.

  COMMAND
    activate            Activate a conda environment.
    build               Build conda packages from a conda recipe.
    clean               Remove unused packages and caches.
    compare             Compare packages between conda environments.
    config              Modify configuration values in .condarc.
    content-trust       Signing and verification tools for Conda
    convert             Convert pure Python packages to other platforms (a.k.a.,
                        subdirs).
    create              Create a new conda environment from a list of specified
                        packages.
    deactivate          Deactivate the current active conda environment.
    debug               Debug the build or test phases of conda recipes.
    develop             Install a Python package in 'development mode'. Similar
                        to `pip install --editable`.
    doctor              Display a health report for your environment.
    index               Update package index metadata files.
    info                Display information about current conda install.
Applications            Initialize conda for shell interaction.
```

And now I finally had hope of being able to install the bcbio-nextgen package.

...but then I realized I still couldn't see the Anaconda Navigator as a GUI version at this point. I saved the state of my VM and restarted it...and quickly realized that this isn't a restore point like in Windows, but a genuinely frozen point in time with all open apps, so more like a hibernation in Windows. So I then truly restarted it from the VirtualBox menu.

I then researched and found that Anaconda Navigator does not show up as an app in the Ubuntu Apps, but can still be opened by running the following very simple command in the terminal:

```
anaconda-navigator
```

And yes, this opened the software! (after some warnings relating to GNOME and rescaling) I chose to update as recommended. However, I did not find the bcbio-nextgen package listed among the packages that could be installed, in spite of creating environments

I discovered that these issues may also be fixed if using mamba as a dependency solver instead of conda. I read the docs of how to install Mamba here. It seems I need to install Miniforge, which seems to be the most recent version, as opposed to using Mambaforge. I read that I must install Miniforge (Mamba) in the so-called base environment and that only it and conda, and no other packages whatsoever, may be installed in the base, because other packages could break both environment managers.

I downloaded the Miniforge .sh binary and ran it in the terminal with `bash Miniforge3-Linux-x86_64.sh`.

It worked:

```
(base) iweber@iweber-VirtualBox:~$ mamba
usage: mamba [-h] [-v] [--no-plugins] [-V] COMMAND ...

conda is a tool for managing and deploying applications, environments and packages.

options:
  -h, --help            Show this help message and exit.
  -v, --verbose         Can be used multiple times. Once for detailed output,
                        twice for INFO logging, thrice for DEBUG logging, four
                        times for TRACE logging.
  --no-plugins          Disable all plugins that are not built into conda.
  -V, --version         Show the conda version number and exit.

commands:
  The following built-in and plugins subcommands are available.

  COMMAND
    activate            Activate a conda environment.
    clean               Remove unused packages and caches.
    compare             Compare packages between conda environments.
    config              Modify configuration values in .condarc.
    create              Create a new conda environment from a list of specified
                        packages.
    deactivate          Deactivate the current active conda environment.
    doctor              Display a health report for your environment.
    info                Display information about current conda install.
    init                Initialize conda for shell interaction.
    install             Install a list of packages into a specified conda
                        environment.
    list                List installed packages in a conda environment.
    notices             Retrieve latest channel notifications.
    package             Create low-level conda packages. (EXPERIMENTAL)
    remove (uninstall)
                        Remove a list of packages from a specified conda
                        environment.
    rename              Rename an existing environment.
```

Followed installation instructions from the bcbio-nextgen page: https://bcbio-nextgen.readthedocs.io/en/latest/contents/installation.html#automated

First, installed Git from git-scm using `sudo apt-get install git`. Checked if I have tar installed by simply typing `tar` in my terminal, and yes, I do.

I then started the installation by calling python3 and, using it, the Py script that is supposed to install bcbio-nextgen. I used the options --nodata to do a minimal installation, without genomes etc, and --mamba to use mamba as a package manager. The command looked like:

```
python3 bcbio-nextgen-install.py /home/iweber/bcbio --tooldir=/home/iweber/bcbio/tools --nodata --mamba
```

I noticed it said it was installing mamba, and am not sure why - maybe because I am not installing bcbio within the mamba directory? But I thought that that amounts to installing in the base environment and is not recommended? Either way, the installation did not succeed. It also told me I was using a deprecated version of conda, so I updated it to the last version (24.3 at the time of writing this).

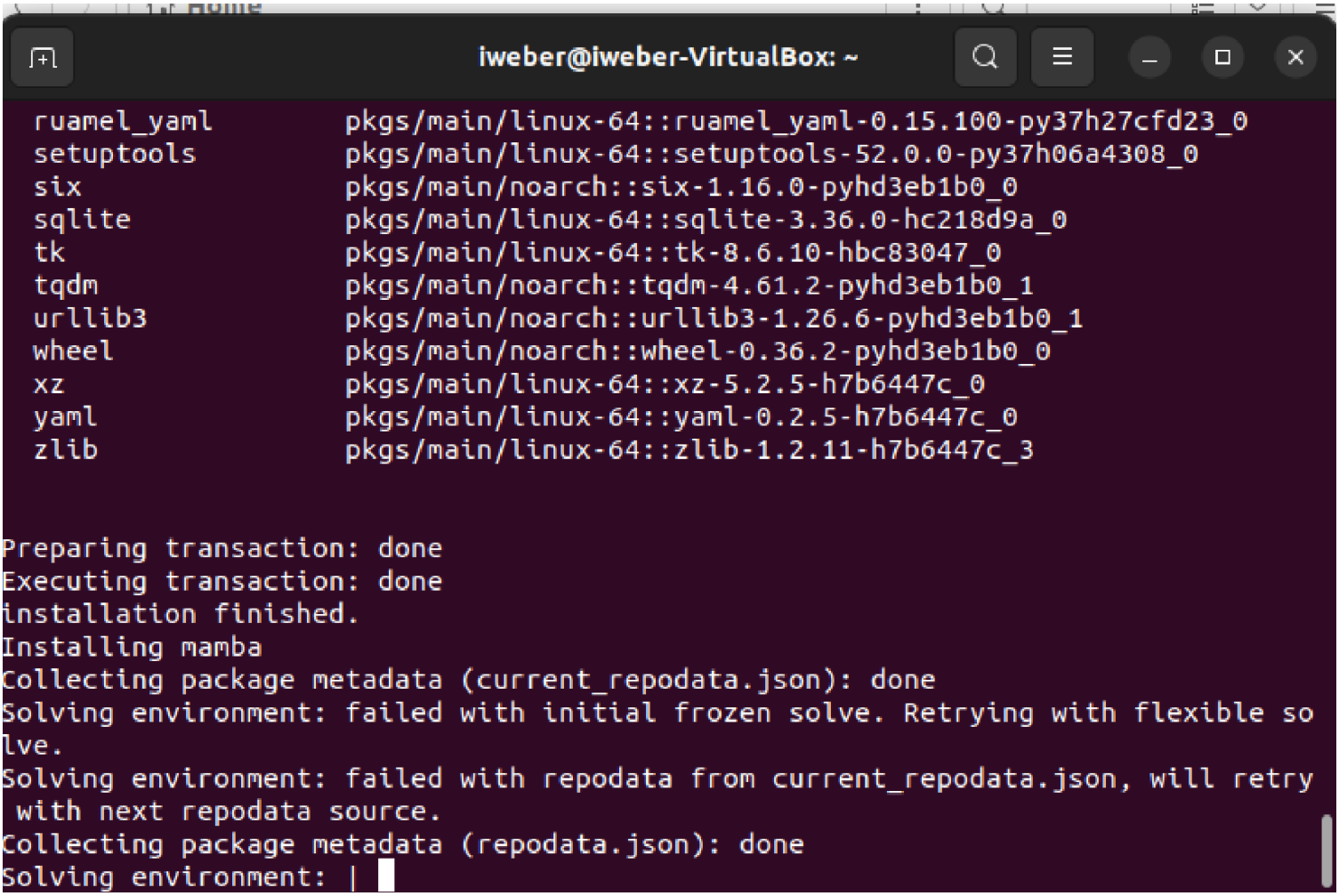
LE: I think the Py installer script for bcbio-nextgen installs Miniconda and uses that to solve the environment/package dependencies, and that's why it pops up as deprecated, even though I just installed the most recent version of conda

Can I somehow force bcbio installer to use the properly installed anaconda3 instead of Miniconda as a solver? - maybe not a good idea. If installer calls for a particular version of conda, it may have a good reason in terms of package dependencies.

After this, whenever I tried running the installation script again, it told me it cannot create the data directory required for the installation, a positional argument that needs to be given upon running the script. So I decided to run it with root privileges using

```
sudo python3 bcbio-nextgen-install.py /bcbio-data --nodata --mamba
```

in order to create a fully new data directory called bcbio-data, and, as before, to not download any genome data and to use mamba as an environment solver instead of conda. And now it seems like I'm stuck in the same solving environment step as has happened every time before when I tried it in the WSL:



I killed the process with `Ctrl+C`. I decided to create a separate conda environment called `bcbio-env` and, within it, to install one of the latest stable versions of Python to see if that helps matters. I had Python 3.10.12 and updated to 3.12.2.

Then, attempted the installation with root privileges in this environment with the same sudo command as before. Same problem with "`failed with intial frozen solve. Retrying with flexible solve.`".

After that, "`failed with repodata from current_repodata.json, will retry with next repodata source`"

Downgraded Python in my bcbio-env to python 3.9.0 and tried installing again with the sudo command. Also "`failed with intial frozen solve. Retrying with flexible solve.`". Killed process.

Upgraded Py to 3.12.2. Tried installing bcbio using pip3 ([pip is the Python package manager]). It said it worked, but, when calling `bcbio_nextgen.py --version`, it gave me an error message saying that the "six" package was not installed. I installed it manually with pip3 and tried calling the version of the package...and then got an error saying another package, toolz, was not installed. Did that as well, only to end up with another package error, this time for "yaml". However, when trying to install yaml via pip3, I got an error mesage that "no version could be found that satisfies requirements/no matching distribution found for yaml."

I ran `mamba clean -a` to remove all of these packages.

As the last strategy, I tried installing bcbio from bioconda as described on the anaconda page:
`conda install bioconda::bcbio-nextgen`

And, suddenly, I could not only use `which bcbio-nextgen` to see where it is installed (/home/iweber/miniforge3/envs/bcbio-env/bin/bcbio_nextgen.py) but ***even got a version number*** when looking for it with `bcbio_nextgen.py --version`, namely version `1.2.9`...which, indeed, is the latest version. At this point, I could've cried of happiness.

As life often goes, I realized only moments later that I can't actually download a new genome because the installation probably didn't create the correct folders for the data that bcbio-nextgen needs to run....

(Side note #2: another thing to remember is to start writing a logfile of everything I do in the terminal. For this, every time I start terminal, should run

`script logfile`

This automatically closes and creates that logfile when exiting terminal with Ctrl+D.

To get all packages currently installed in my conda environment:

`conda list --export > my_conda_packages.txt` [what was the point of this?]

I closed the VM at this point to come back to it another day...only to come back to not being able to open the Anaconda Navigator anymore from bash...

I added the path to the anaconda binaries folder to my PATH variable in the .profile file so that the bash knows to look here for executables as well, because I'd like to type less whenever I want to start the Anaconda Navigator: `export PATH=$PATH:/home/iweber/anaconda3/bin`. This, however, did not solve the issue.

I went ahead and updated the installation with `bash Anaconda3-2024.02-1-Linux-x86_64.sh -u
and got yet another weird error message.

And...the same errors over and over again. I looked further into this pipeline and decided its maintenance is lagging too far behind, as, for many other bioinformatic projects, the main contributors have, in the meanwhile, moved on to other projects.

I have decided to, instead, use Nextflow and established NGS data analysis pipelines that work on it.

Fixing Win-Linux clipboard

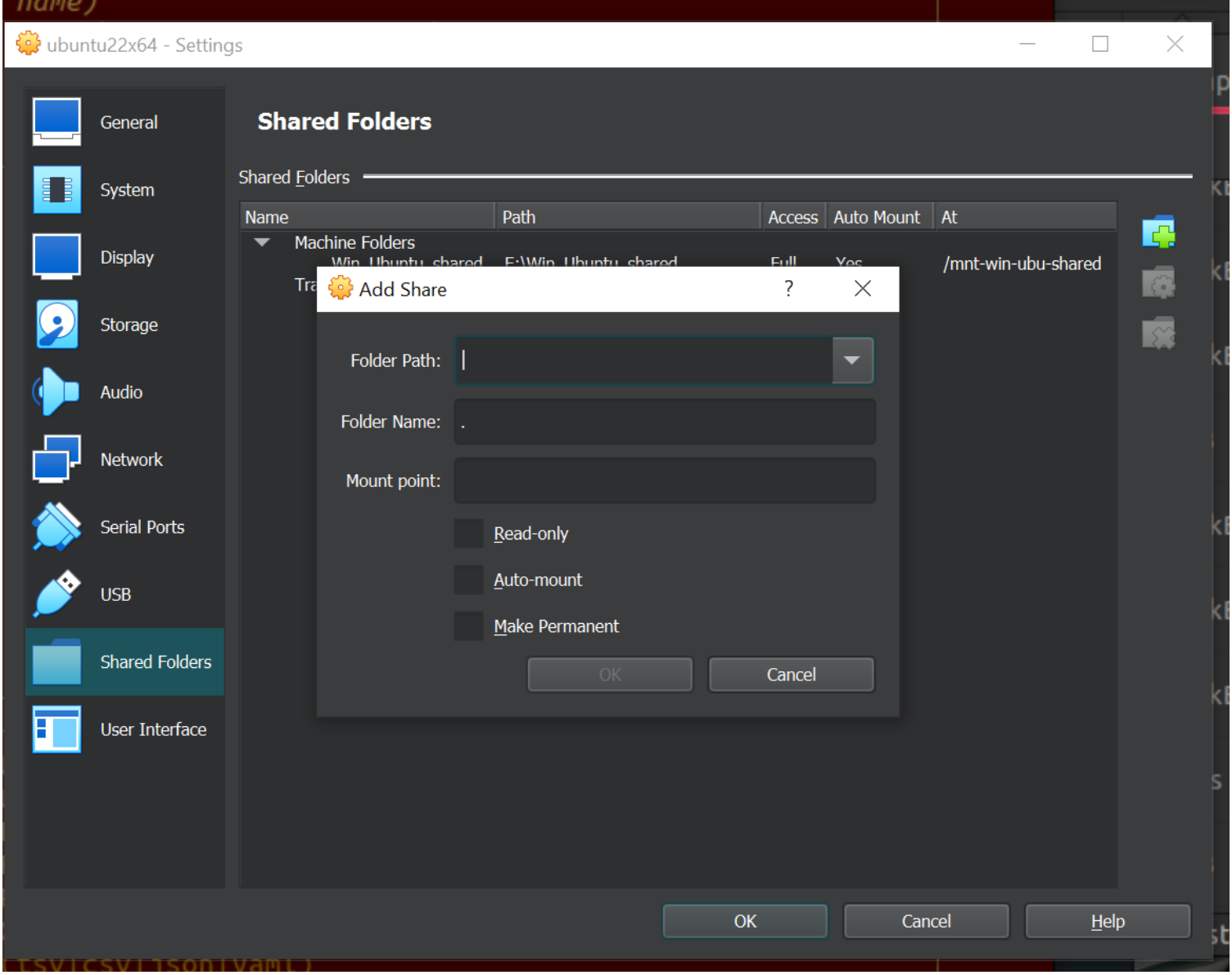
In the meanwhile, I tried fixing my highly annoying copy-paste and drag-and-drop issues between my virtual machine and my Windows host. I first installed:

`sudo apt-install build-essential dkms linux-headers-generic`,

following this answer and the page it refers to . I "mounted" the VA Guest Box addition disc image and ran the "autorun.sh" bash script it contains as a program (right click -> "Run as program"). After it asked for credentials, the terminal opened and I saw the expected screen. Restarted Ubuntu et voila! Copying and pasting now finally works.

Creating an inter-OS shared folder between Win and Linux

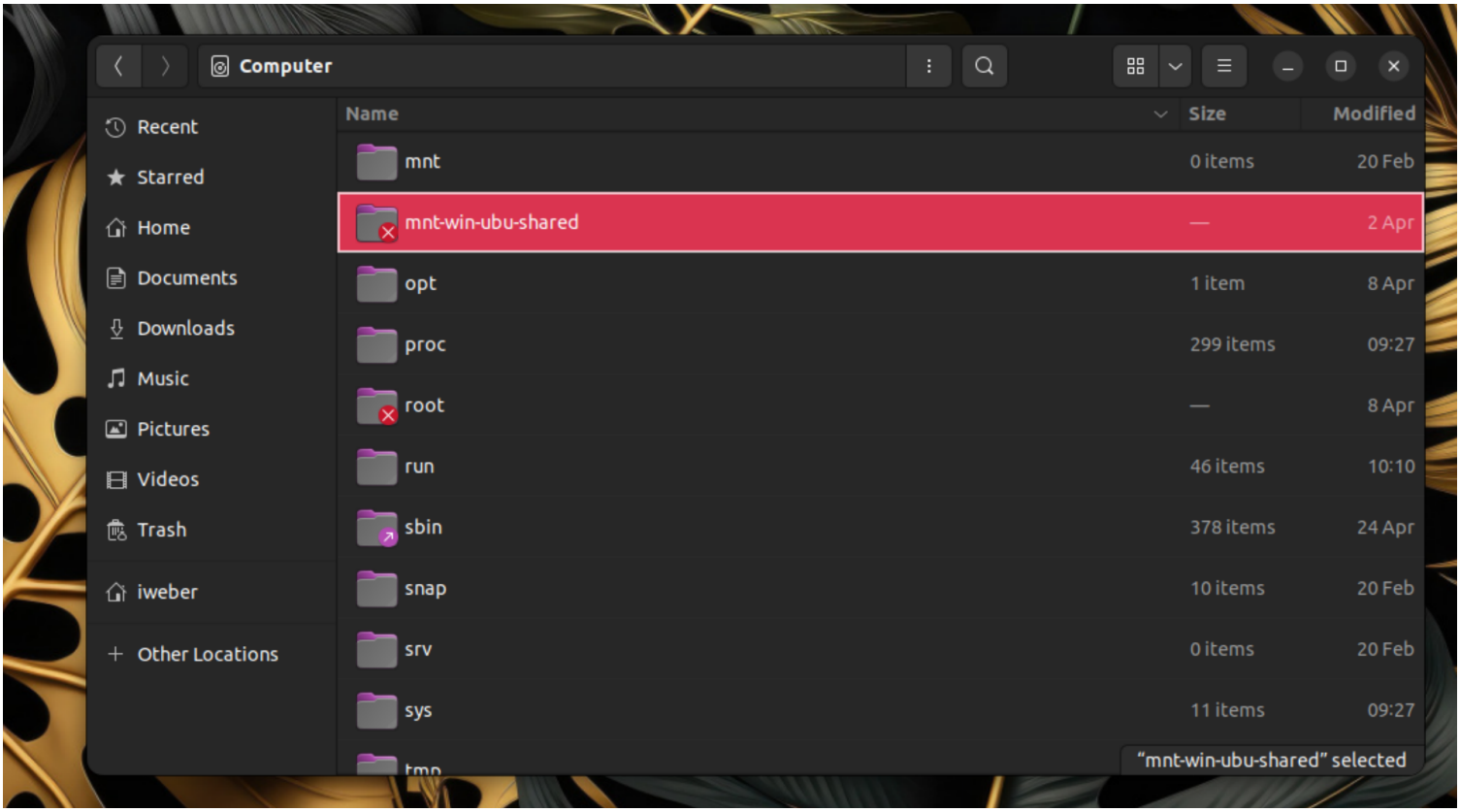
I had previously created a shared folder between my Windows host system and the Ubuntu virtual machine using the VM menu "Devices" -> "Shared folders" -> "Shared folders settings". Clicking the blue folder icon with a plus sign on it, I got such a window:



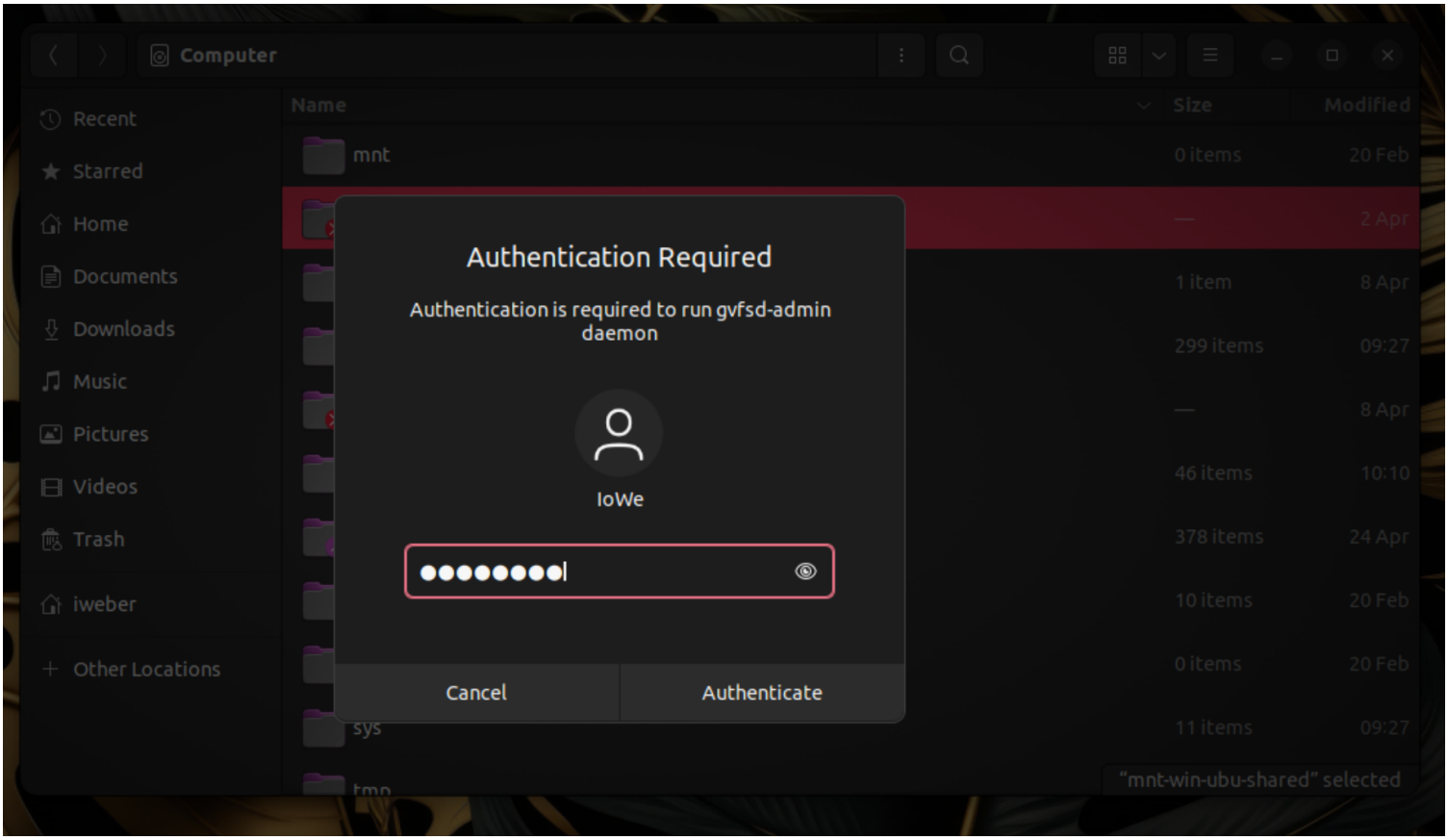
The "Folder Path" field allowed me to select a folder in my host to use as the shared folder. For this purpose, I went to Windows and, on the partition of the SDD that I had allotted to the virtual machine, I created a folder called "Win_Ubuntu_shared". I could then access this from the VM menu displayed above, and selected to auto-mount it and make it permanent in order to have it automatically accessible whenever I run the VM. I also gave it a mount point name in order to find it more easily under Linux, and chose "/mnt-win-ubu-shared" for this name.

But I didn't see this folder in my Linux file explorer, not even after restarting the virtual machine. I followed the instructions given here and ran `sudo usermod -aG vboxsf $USER` in terminal, and then tried accessing folder via Other locations -> Computer, where I finally saw the "mnt-win-ubu-shared"

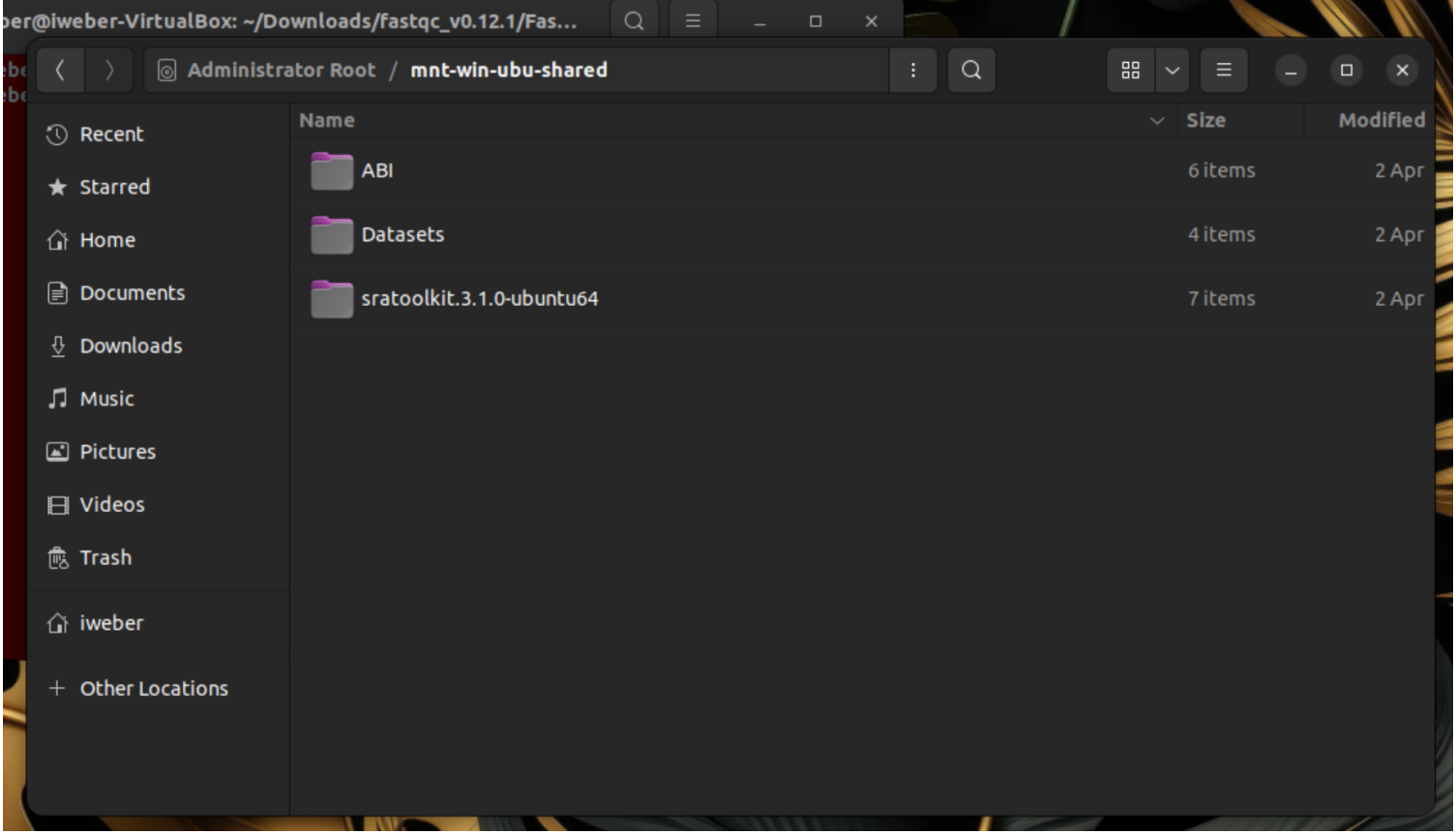
folder (remember, this is the name I gave the folder when setting it up in the virtual machine in the VirtualBox menu "Devices"> "Shared folders").



Got a message window asking me to also log in with my user credentials.



I did so, and I could finally see the contents of the shared folder! I immediately bookmarked this mnt-win-ubu-shared folder from the address bar of the explorer (three-dot menu) for easy access later.



I could now *finally* access the FastQ files I had previously generated on the WSL for further analyses!

Making the inter-OS shared folder auto-mount without asking for my password every time

I had previously created a folder for the internship data that's shared between my Windows host OS and Linux , and I realized I wasn't using it as the sole folder for my data in Linux because, every time I wanted to access it, it asked me for my user password. That made me reluctant to use it especially in the context of an automated analysis pipeline, because I feared that Linux suddenly forgetting the password might interrupt the pipeline at inconvenient times, e.g. when I'm not at my laptop. Additionally, this complicated setting up a proper GitHub repo. So, to fix that, I asked ChatGPT.

I used `mount | grep mnt-win-ubu-shared` to check where the folder is located. It returned `Win_Ubuntu_shared on /mnt-win-ubu-shared type vboxsf (rw,nodev,relatime,iocharset=utf8,uid=0,gid=999,dmode=0770,fmode=0770,tag=VBoxAutomounter)`, so I now knew the address of my mounted folder.

I added my user to the permissions group for the virtual box folder (`vboxsf`) with `sudo usermod -aG vboxsf $USER`, then ran a `chown` command to change ownership of the folder:
`sudo chown -R $USER:vboxsf /mnt-win-ubu-shared/

Now, when checking the permissions, I saw:

```
(base) iweber@iweber-VirtualBox:~$ mount | grep mnt-win-ubu-shared
Win_Ubuntu_shared on /mnt-win-ubu-shared type vboxsf (rw,nodev,relatime,iocharse
t=utf8,uid=0,gid=999,dmode=0770,fmode=0770,tag=VBoxAutomounter)
(base) iweber@iweber-VirtualBox:~$ ^C
(base) iweber@iweber-VirtualBox:~$ sudo usermod -aG vboxsf $USER
[sudo] password for iweber:
(base) iweber@iweber-VirtualBox:~$ sudo chown -R $USER:vboxsf /mnt-win-ubu-share
d/
(base) iweber@iweber-VirtualBox:~$ ls -l /mnt-win-ubu-shared/
total 8
drwxrwx--- 1 root vboxsf 4096 Mai 22 13:40 ABI
drwxrwx--- 1 root vboxsf    0 Apr  2 15:22 Datasets
drwxrwx--- 1 root vboxsf 4096 Apr  2 15:23 sratoolkit.3.1.0-ubuntu64
(base) iweber@iweber-VirtualBox:~$
```

so I knew I now have read and write privileges for the directory for both myself and the virtual box group I am in. I copied and pasted some files into this folder from both Windows and Linux to make sure it works bidirectionally, as it should, and it does.

Giving the inter-OS shared folder the correct permissions upon startup

However, upon restarting the virtual machine, Ubuntu still asked me for my password (specifically, "Authentication is required to run gvfsd-admin daemon") when trying to open the inter-OS shared folder...ChatGPT said this is likely due to startup permissions not being set properly and suggested I edit a file called "fstab" with admin privileges and add the path to the inter-OS shared folder. "fstab" is a "file systems table" with information about files, folders, and mount points [other source to confirm?]. I added `Win_Ubuntu_shared /mnt-win-ubu-shared vboxsf uid=1000,gid=1000,dmode=0770,fmode=0770 0 0` at the end of the file. I tried it, restarted the system, and I still got asked for the password...

ChatGPT's next idea was to create a script that will be automatically run at system startup and will mount the inter-OS shared folder with the appropriate permissions from the get-go. I removed the line I previously added to the fstab file and proceeded to create the new startup file. I created the file directly with nano, `sudo nano /usr/local/bin/mount_shared_folder.sh`, and added to it

`#!/bin/bash` sudo mount -t vboxsf -o uid=1000,gid=1000,dmode=0770,fmode=0770 Win_Ubuntu_shared /mnt-win-ubu-shared`

I made it executable with `sudo nano /etc/systemd/system/mount_shared_folder.service`, then created a service to run it at startup: `sudo nano /etc/systemd/system/mount_shared_folder.service`, with the contents

```
[Unit]
Description=Mount VirtualBox Shared Folder
After=vboxadd.service
Requires=vboxadd.service

[Service]
Type=oneshot
ExecStart=/usr/local/bin/mount_shared_folder.sh
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
```

.
Then, reloaded the Systemd and enabled the newly created service:

```
sudo systemctl daemon-reload
sudo systemctl enable mount_shared_folder.service
```

, which returned

```
Created symlink /etc/systemd/system/multi-user.target.wants/mount_shared_folder.service →
/etc/systemd/system/mount_shared_folder.service.
```

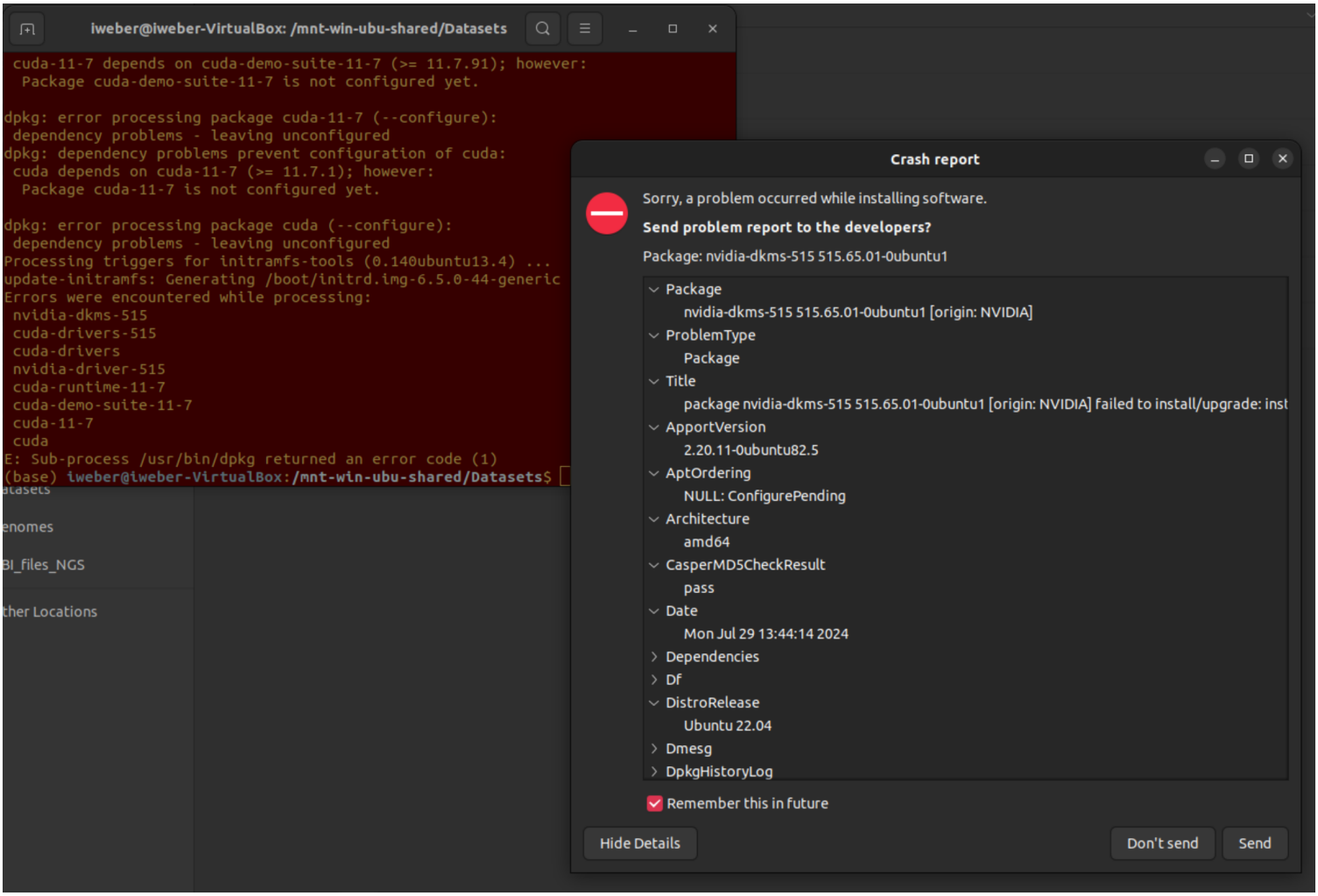
And I restarted the service with `sudo systemctl start mount_shared_folder.service`, and then the virtual machine itself. And it still doesn't work without punching in my password. So I decided that, for the time being, if I only have to input the password once, at system startup, that's fine and unlikely enough to affect the pipeline that I'm willing to drop it and move onto something more productive.

I added aliases to my .bashrc file to more easily access the folder, if nothing else. I just pasted into it:

```
alias inter-OS='cd /mnt-win-ubu-shared'
alias inter-OS_data='cd /mnt-win-ubu-shared/Datasets/'
```

Additionally, I installed conda environment name autocompletion. I first tried installing the bash autocompletion just to be sure it works properly with `sudo apt-get install bash-completion

, but that ran into an error:



The error seems to be related with the Nvidia and Cuda drivers...and I don't have the time to look further into this right now.

I simply proceeded with `conda install bash-completion` in the base environment. It seems to have completed error-free, and the autocompletion now works without errors.

Updating installation for the SRA toolkit

I had initially worked on the WSL and created the VirtualBox virtual machine later, so of course the SRA toolkit was not working any longer. I created a dedicated conda environment for it called `ncbi_SRA`, and installed the package from Bioconda using `conda install sra-tools` and updated it with `conda update sra-tools`

Creating a GitHub repository for the project

In order to be able to more easily version my work on the project and to allow my internship supervisor to gain insight into what I was doing, I created a GitHub repo for the newly minted "official project folder" (the shared folder I created above).

I first created an empty repo on GitHub and also a fine-grained access token for myself, giving myself all of the privileges necessary for using the repo. Then, from Ubuntu, I initialized a new git repo in the inter-OS shared folder with `git init` and pointed its head to the main branch: `git symbolic-ref HEAD refs/heads/main`.

I created the link to the online repo with
`git remote add origin https://github.com/i-weber/Internship_project_RNAseq`.

I added all of the files in the folder to the current staging area with `git add .`, and that might have been a mistake, given that the FastQ files are pretty large in their uncompressed state - the whole folder is around 170 GB *insert clenched teeth smiley*. Indeed, I then found out one can use the `top` command ***in a new terminal window*** to check resource usage in Linux, similar to Windows's task manager, and I saw that Git is using 70-80% of all resources of the virtual machine, so I gave it time. Next time, I will use

```
git add . --verbose
```

to get a better feel of the progress it is making through the files and folders. Also, can use `iostat -x 1` to investigate disk usage (install with `sudo apt-get install sysstat`) or `htop`, which apparently is `top` on steroids (also needs prior installation with `sudo apt-get install htop`).

So far, clocking at half an hour run time for the add command...let's see how long it takes in total.
At the one hour mark, I decided it was time to stop messing around, so I killed the process and proceeded to add the FastQ files and genomic files to

the .gitignore. I created it with nano and added to it:

```
Adapters/
Genomes/
Nextflow/
Software/
Datasets/sra
Datasets/Pre_eclampsia_mice/Pre_eclampsia_mice_fastq/
```

to avoid some of the largest and recoverable files from the push and commit.

I then tried to push the .gitignore only, and , after completing the command line sign in, immediately got an error because I had previously set GitHub to block commits that might publicize my email address. So I found out that, in order to avoid this issue, I can use a no-reply address that GitHub itself creates for all of its users as my default email. I ran `git config --global user.email "163516184+i-weber@users.noreply.github.com"` to do so, and deactivated the checkbox in my GitHub email setttings that supposedly protects one from publicizing their email in the commit metadata (I now know that this no-reply address will be used). And I could finally push the .gitignore file and see it on the website!

I then proceeded to add all other files with `git add . --verbose` at 16.48. It crashed around 17.05 because - and I should've thought of this! - I deleted a file from the folder at some point ***facepalm***. One more thing learned...and restarted the same command at 17.06. Annnd it worked!

I proceeded to commit the changes with `git commit -m "All files up to date"`, which returned

```
[main 3a13867] All files up to date
 131 files changed, 98102 insertions(+), 1 deletion(-)
```

It failed, with an error message saying: "Enumerating objects: 129, done. Counting objects: 100% (129/129), done. Delta compression using up to 6 threads Compressing objects: 100% (123/123), done. error: unable to rewind rpc post data - try increasing http.postBuffer error: unable to rewind rpc post data - try increasing http.postBuffer error: RPC failed; HTTP 400 curl 92 Recv failure: Connection reset by peer send-pack: unexpected disconnect while reading sideband packet Writing objects: 100% (127/127), 2.04 GiB | 2.85 MiB/s, done. Total 127 (delta 18), reused 1 (delta 0), pack-reused 0 (from 0) fatal: the remote end hung up unexpectedly Everything up-to-date"

I set the postBuffer to 200 MB using `git config --global http.postBuffer 209715200` , and re-started the commit command at 23:20.

Aaand...it failed again. The message read just as above (I think - I lost the clipboard because I shut the virtual machine off too soon, apparently...)

New environment, new life: creating dedicated bioinformatics environments using conda

Since I had all of the aforementioned issues with bcbio, I decided to abandon that approach, and, consequently, deleted the environment with conda, just to be on the safe side that I will have no issues later:

`conda env remove --name bcbio-env`

I then created a brand new, fresh environment called "NGS" in which to install the QC packages:
`conda env --create NGS`. Made a separate conda environment called "NGS" with `conda create --name NGS`

Using the shared folder between my Windows host and my Ubuntu virtual machine, I copied the FastQ files I had previously generated with WSL into the virtual machine.

Installing FastQC on my NGS conda environment within the Linux virtual machine

Perl and Java were already installed on my system (I suspect I installed them when installing conda and it creating its base environment), which is important because FastQC needs to run a small Perl script to find the binaries (executable files). The rest was as easy as 123: downloaded the archive, unzipped it, made the file called "fastqc" executable (chmod u+x) and ran it with ./fastqc.

Added path to the FastQC folder to my PATH environment variable so that I can now only type `fastqc` from any folder and Linux still finds the binaries of fastqc and can open it:

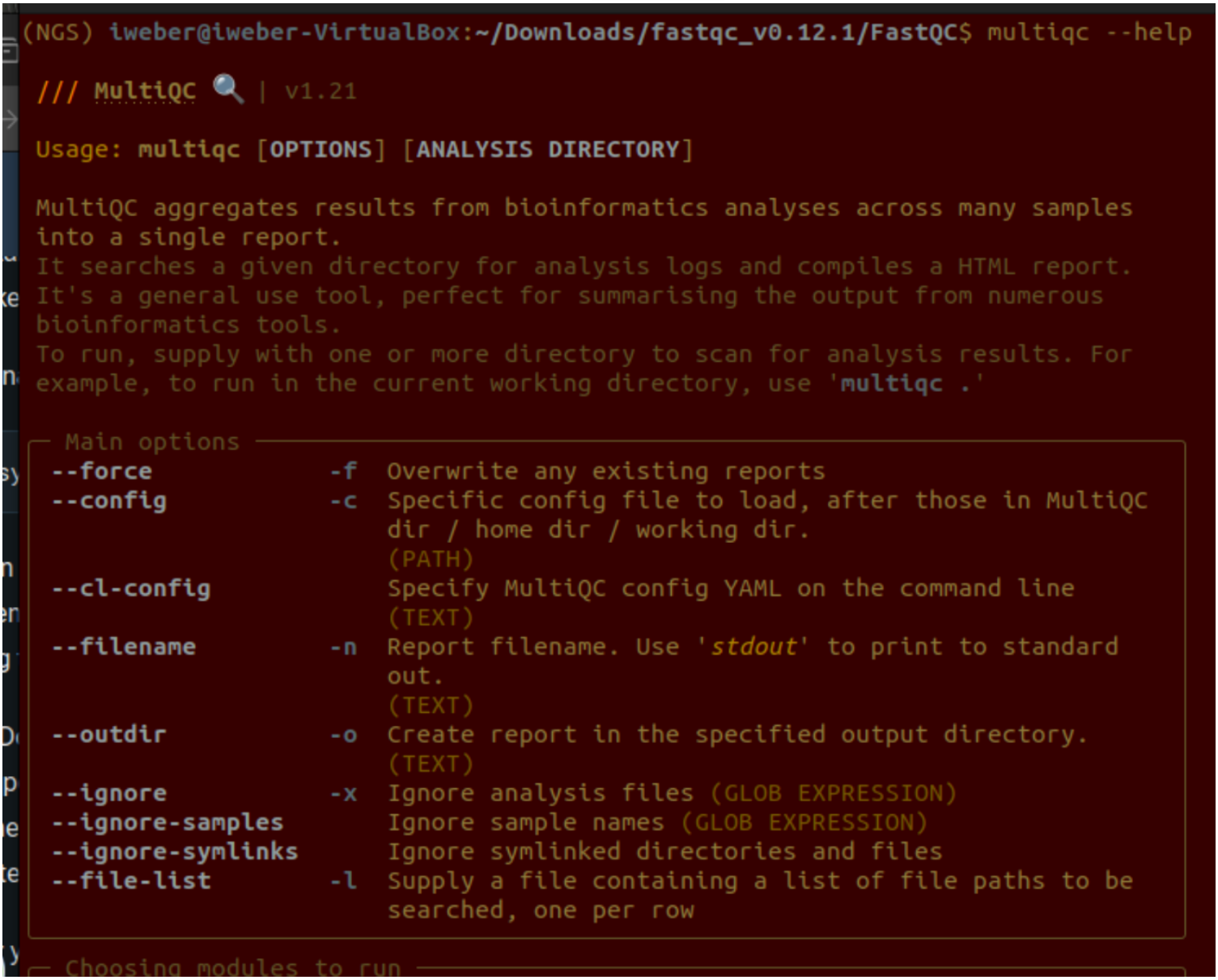
`sudo ln -s /home/iweber/Downloads/fastqc_v0.12.1/FastQC/fastqc /usr/local/bin/fastqc`

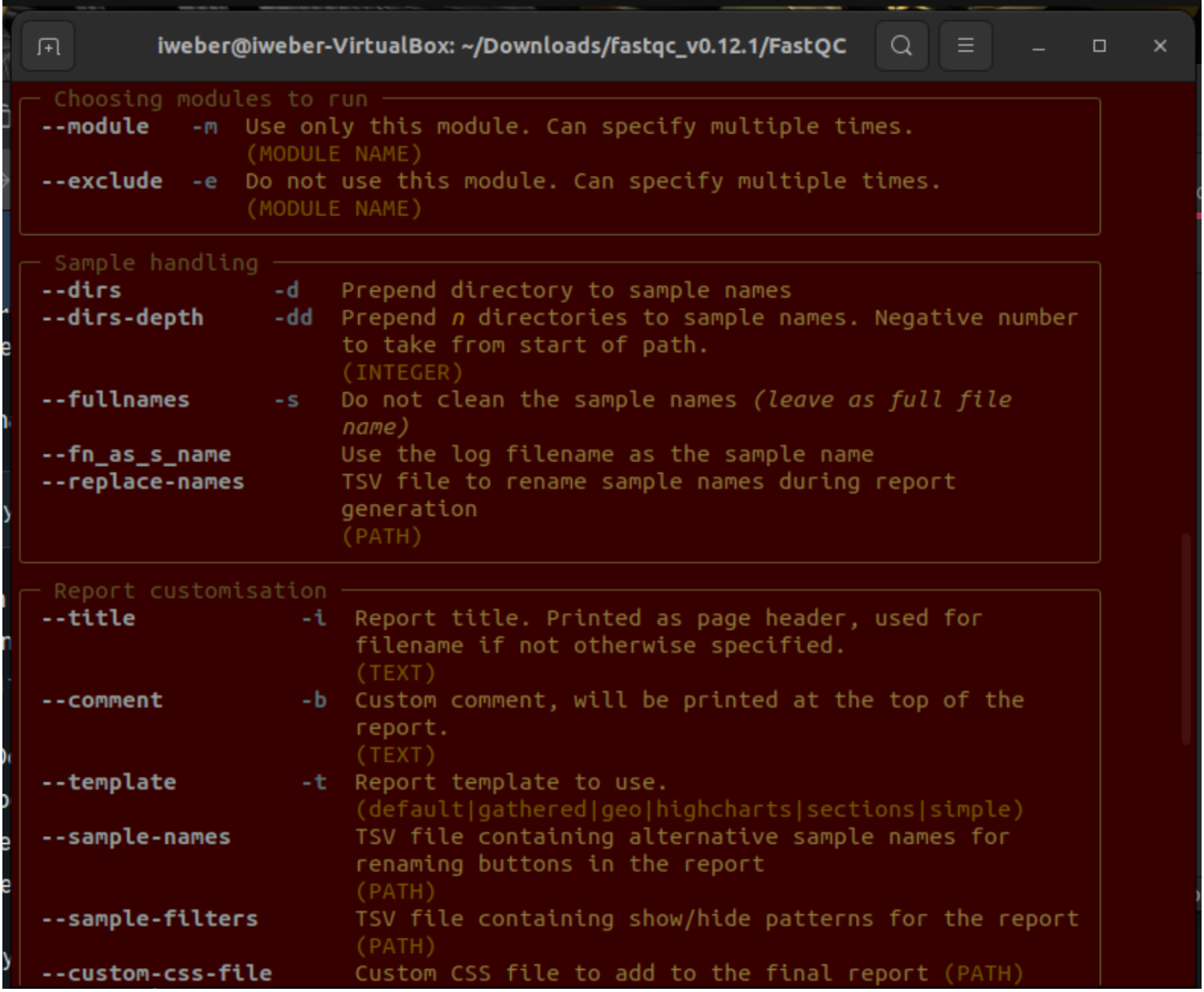
This worked! I can now call `fastqc` from the command line.

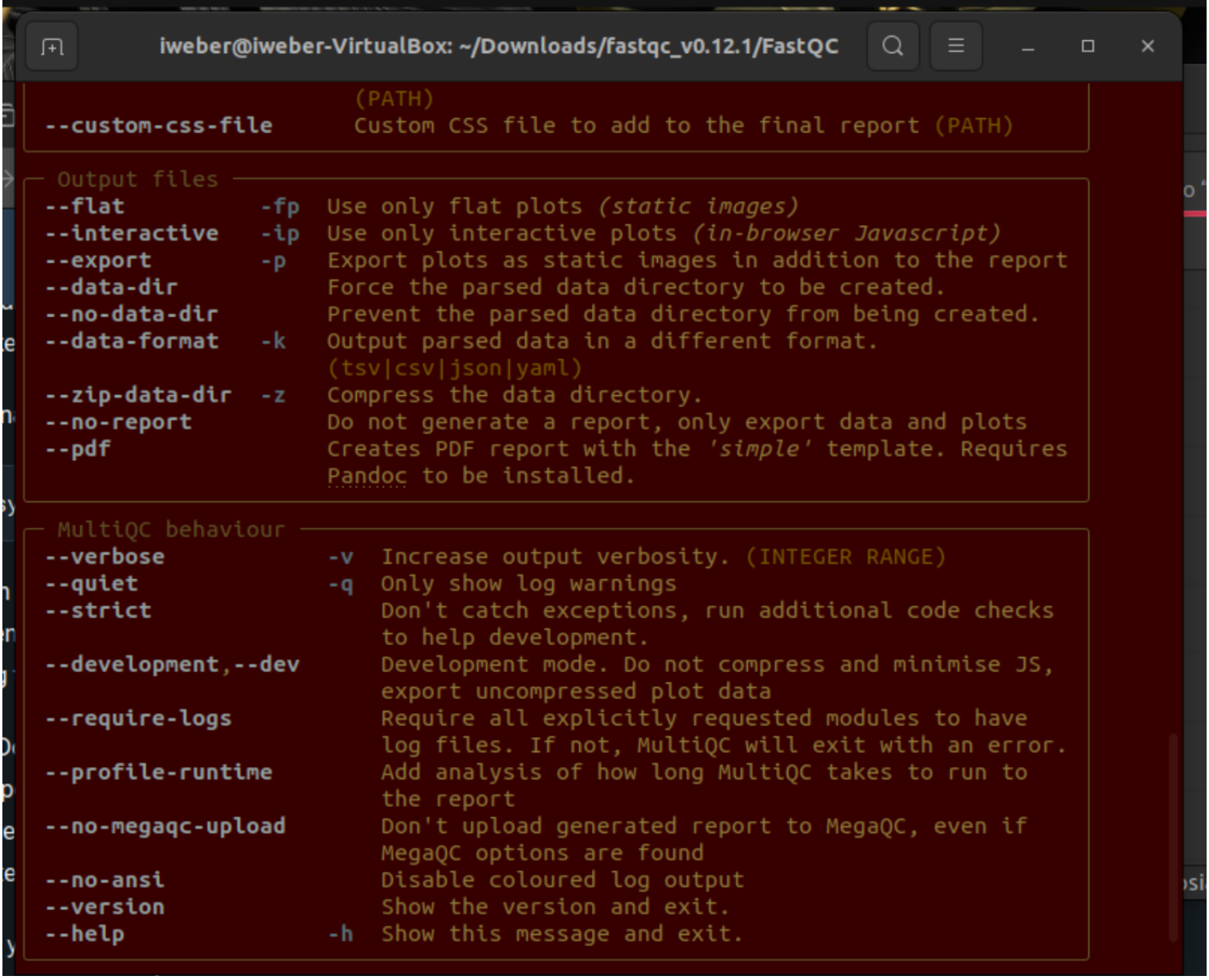
Installing MultiQC on my NGS conda environment within the Linux virtual machine

Once more, this was extra simple from the command line:
`conda install bioconda::multiqc` , after which I could call it from the command line.

After that, I wanted to see what options are available for multiqc, so I ran `multiqc --help`. I got quite a number of options:




iweber@iweber-VirtualBox: ~/Downloads/fastqc_v0.12.1/FastQC
Choosing modules to run
--module -m Use only this module. Can specify multiple times. (MODULE NAME)
--exclude -e Do not use this module. Can specify multiple times. (MODULE NAME)
Sample handling
--dirs -d Prepend directory to sample names
--dirs-depth -dd Prepend n directories to sample names. Negative number to take from start of path. (INTEGER)
--fullnames -s Do not clean the sample names (leave as full file name)
--fn_as_s_name Use the log filename as the sample name
--replace-names TSV file to rename sample names during report generation (PATH)
Report customisation
--title -i Report title. Printed as page header, used for filename if not otherwise specified. (TEXT)
--comment -b Custom comment, will be printed at the top of the report. (TEXT)
--template -t Report template to use. (default|gathered|geo|highcharts|sections|simple)
--sample-names TSV file containing alternative sample names for renaming buttons in the report (PATH)
--sample-filters TSV file containing show/hide patterns for the report (PATH)
--custom-css-file Custom CSS file to add to the final report (PATH)



Installing Nextflow and nf-core

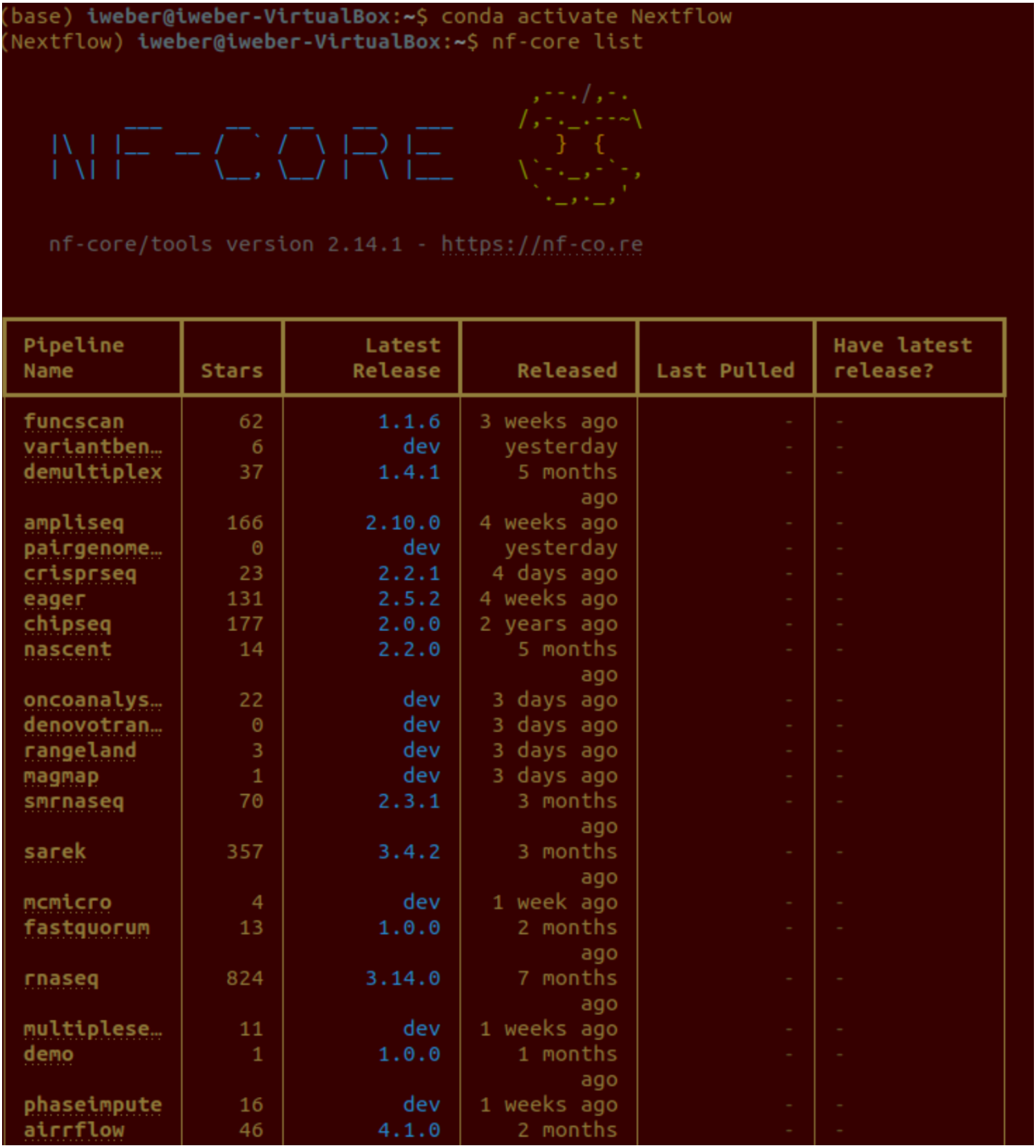
First, installed Java:
`sudo apt install default-jre`

I created a dedicated environment in which i will work with Nextflow, that I called as such, using `conda create Nextflow`.

After activating it, I installed the actual Nextflow workflow manager with `conda install -c bioconda nextflow`, and it worked like a charm. Then, to get access to the curated Nextflow pipelines for bioinformatic analyses, I installed nf-core in the same environment with `conda install nf-core`, which also completed without any overt errors.

I also activated shell completions for nf-core by adding the recommended command to my .bashrc file: `eval "$(_NF_CORE_COMPLETE=bash_source nf-core)"` (https://nf-co.re/docs/nf-core-tools/installation#activate-shell-completions-for-nf-coretools), and restarted my shell.

And now I could easily list all of the curated pipelines available in nf-core:



```
(base) iweber@iweber-VirtualBox:~$ conda activate Nextflow
(Nextflow) iweber@iweber-VirtualBox:~$ nf-core list
```

```
                                          ,--./,-.
          ___     __   __   __   ___     /,-._.--~\
    |\ | |__  __ /  ` /  \ |__) |__         }  {
    | \| |       \__, \__/ |  \ |___     \`-._,-`-,
                                          `._,._,'
```

nf-core/tools version 2.14.1 - https://nf-co.re

Pipeline Name	Stars	Latest Release	Released	Last Pulled	Have latest release?
funcscan	62	1.1.6	3 weeks ago	-	-
variantben…	6	dev	yesterday	-	-
demultiplex	37	1.4.1	5 months ago	-	-
ampliseq	166	2.10.0	4 weeks ago	-	-
pairgenome…	0	dev	yesterday	-	-
crisprseq	23	2.2.1	4 days ago	-	-
eager	131	2.5.2	4 weeks ago	-	-
chipseq	177	2.0.0	2 years ago	-	-
nascent	14	2.2.0	5 months ago	-	-
oncoanalys…	22	dev	3 days ago	-	-
denovotran…	0	dev	3 days ago	-	-
rangeland	3	dev	3 days ago	-	-
magmap	1	dev	3 days ago	-	-
smrnaseq	70	2.3.1	3 months ago	-	-
sarek	357	3.4.2	3 months ago	-	-
mcmicro	4	dev	1 week ago	-	-
fastquorum	13	1.0.0	2 months ago	-	-
rnaseq	824	3.14.0	7 months ago	-	-
multiplese…	11	dev	1 weeks ago	-	-
demo	1	1.0.0	1 months ago	-	-
phaseimpute	16	dev	1 weeks ago	-	-
airrflow	46	4.1.0	2 months	-	-

The pipeline I am interested in is called `rnasplice`, and we'll get back to that in a moment.

Containerization with Docker for nf-core/Nextflow?

What caught my eye when reading more about nf-core was the following:

Pipeline software

An analysis pipeline chains the execution of multiple tools together. Historically, all tools would have to be manually installed — often a source of great frustration and a key step where reproducibility between analyses is lost. nf-core pipelines utilise the built-in support for software packaging that Nextflow offers: all can work with Docker and Singularity, and most pipelines also support Conda.

To use any of the below, simply run your nf-core pipeline with `-profile <type>`. For example, `-profile docker` or `-profile conda`.

- Docker
  - Typically used locally, on single-user servers, and the cloud
  - Analysis runs in a *container*, which behaves like an isolated operating system
  - Typically requires system root access, though a *"rootless mode"* is available
- Singularity
  - Often used as an alternative to Docker on HPC systems
  - Also runs *containers*, and can optionally create these from Docker images
  - Does not need root access or any daemon processes
- Apptainer
  - Open source version of Singularity (split from Singularity in 2021)
  - **⚠ Warning**

    Currently, nf-core pipelines run with `-profile apptainer` will build using docker containers instead of using pre-built singularity containers.
    To use the singularity containers, use `-profile singularity` instead. This works because `apptainer` simply defines `singularity` as an alias to the `apptainer` command.
- Podman, Charliecloud and Shifter
  - All alternatives to Docker, often used on HPC systems
- Conda
  - Packaging system that manages environments instead of running analysis in containers
  - Poorer reproducibility than Docker / Singularity
    - There can be changes in low-level package dependencies over time
    - The software still runs in your native operating system environment and so core system functions can differ
- Mamba
  - A faster reimplementation of Conda

I started working with conda because everyone in the bioinformatic community swears by it, and so did the economics researchers that I had my very first Python course with. It has served me very well so far, but I do also know that software developers prefer to work with Docker, Singularity, or Kubernetes to create containers so that their software can always be run, at any time, on any machine. However, when reading more into how Docker, Conda, and Nextflow relate to one another, I realized it may not make sense to work with Docker in my current context. Conda is used so widely in the bioinformatic/scientific community because it integrates seamlessly with Python and R and is a more lightweight solution for reproducibility because it manages only the dependencies specifically required by these programming languages. However, Docker containers also incorporate information about the OS and all of the apps and packages installed on it that are required to run a specific program/application. This increases the degree of reproducibility dramatically BUT is also more bulky, because a Docker container then stores all of this extra information related to the OS.

Quality control of FastQ files with the RNA-seq reads

Quality control of individual FastQ files

For the quality control of this small number of files, I used FastQC, which is an open-source tool that checks the reads in FastQ files for various quality parameters, such as sequence quality scores, GC content, sequence duplication levels, or the presence of sequences left behind by the oligonucleotide adapters used for the amplification and actual sequencing of the cDNA fragments. I was pleasantly surprised that opening the application from the Ubuntu command line resulted in a user-friendly GUI, with which I could select the sequences to be analyzed. After analyzing the two FastQ files resulting from the first sequencing run (SRR13761520_1 and SRR13761520_2), I got quite different results. Whereas the file ending in 2 passed the quality checks in all but one department ("Per base sequence content" gave a warning), the file ending in 1 had a few more issues. It failed the "Per base sequence content" testing and gave warnings for "Per sequence GC content" and "Sequence duplication levels". I suppose this difference results from one of the files representing one item from each pair of reads and the other file the sequence complementary to it [source?]. As these two reads stem from two separate sequencing reactions [source], it is normal that one set can have different sequencing quality, based on differences in the reaction setup, base composition, or technically-introduced biases that only happened for one of the two sequencing runs.

Nonetheless, I wanted to understand better what the parameters returned by FastQC are and what they mean for the usability of the data for further analyses. To better understand how this data was pre-processed, I foraged a bit in the Series Matrix file provided on the GEO page of the study, but the authors don't give a lot of details about the handling of the data, aside from `Reads filtering under criteria removing reads with 20% of the base quality lower than 13"` . Considering that FastQC did not indicate the presence of any known adapter sequences in the data and that the files were already clearly sorted by experimental condition and subject, (I hope) it's safe to assume that any necessary demultiplexing and adapter trimming has been performed by the authors of the study/the sequencing facility they worked with, plus applying this quality filter they mention in the Series Matrix file.

To centralize all analysis info for all of the FastQ files generated by FastQC, I settled on MultiQC . I envision myself working with single-cell RNA-seq datasets at a later time point, and MultiQC can, as the name suggests, run the analyses in parallel on many files, making it a lot more efficient in handling large numbers of datasets. MultiQC is a part of the bcbio-nextgen Python package, so I wanted to install it in my Python on the WSL Ubuntu as instructed by the package's GitHub page. I thought this installed not only the package but also all dependencies (other modules) that are needed for it to run...but it didn't. It told me it was missing crucial data, such as the genome assemblies for mouse and human, which prevented it from running at all.

During the lengthy installation process, I kept getting a message for all packages "Solving environment: failed with initial frozen solve. Retrying with flexible solve.". Thanks to another kind internet stranger, I found that this is a problem that stems from having a version of Python that's newer than what many of these packages are optimized to work with. [what version do i have on my WSL? what versions needed? conda search python, then conda install python=desired_version_number, and then restart Ubuntu and Py, and update bcbio-nextgen package, hoping that this will resolve dependency issues]. However, the installation of bcbio-nextgen was still running (also, still running after 3h with the WSL virtual machine using 85 GB RAM sounds unusual and a bit alarming).

I realized that this may be due to WSL being less efficient in its memory usage than a full-fledged virtual machine [source?], so I proceeded to kill this process and install Oracle VirtualBox 7 and make an Ubuntu VM on it.

MultiQC: General quality stats of the pre-eclampsia FastQ files

I saw that multiqc takes one full folder as input. I could call `fastqc` from the command line individually for all of my FastQ files, but...why would I do something rather tedious to do for all of the 16 FastQ files? So I quickly wrote a small Bash script to automate the process:

`#!/bin/bash

`directory="/home/iweber/Documents/ABI_files_NGS/6484_fastq/"

`for file in "$directory"/*.fastq; do\if [-f"file"]; then
`fastqc "$file" --outdir /home/iweber/Documents/ABI_files_NGS/FastQC_results` fi

`done`

This simply cycles through all of the files with a .fastq ending in the indicated directory, stores the name of the file in the variable called file, and then calls fastqc to operate on the content of the file name variable ($file), while outputting the result into the FastQC_results folder.
I made it executable with `chmod` and then ran it, and:

```
(NGS) iweber@iweber-VirtualBox:~/Documents/Datasets/Pre_eclampsia_mice$ ./fastqc_loop_folder.sh
null
Started analysis of SRR13761520_1.fastq
Approx 5% complete for SRR13761520_1.fastq
Approx 10% complete for SRR13761520_1.fastq
Approx 15% complete for SRR13761520_1.fastq
Approx 20% complete for SRR13761520_1.fastq
Approx 25% complete for SRR13761520_1.fastq
Approx 30% complete for SRR13761520_1.fastq
Approx 35% complete for SRR13761520_1.fastq
Approx 40% complete for SRR13761520_1.fastq
Approx 45% complete for SRR13761520_1.fastq
Approx 50% complete for SRR13761520_1.fastq
Approx 55% complete for SRR13761520_1.fastq
Approx 60% complete for SRR13761520_1.fastq
Approx 65% complete for SRR13761520_1.fastq
Approx 70% complete for SRR13761520_1.fastq
Approx 75% complete for SRR13761520_1.fastq
Approx 80% complete for SRR13761520_1.fastq
Approx 85% complete for SRR13761520_1.fastq
Approx 90% complete for SRR13761520_1.fastq
Approx 95% complete for SRR13761520_1.fastq
Analysis complete for SRR13761520_1.fastq
null
Started analysis of SRR13761520_2.fastq
Approx 5% complete for SRR13761520_2.fastq
```

It worked! Naturally, I then ran MultiQC to summarize all of the reports obtained with FastQC.

```
(NGS) iweber@iweber-VirtualBox:~/Documents/Datasets/Pre_eclampsia_mice$ multiqc FastQC_results/

  /// MultiQC 🔍 | v1.21

|           multiqc | Search path : /home/iweber/Documents/Datasets/Pre_eclampsia_mice/FastQC_results
|         searching | ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 32/32
|            fastqc | Found 16 reports
|           multiqc | Report      : multiqc_report.html
|           multiqc | Data        : multiqc_data
|           multiqc | MultiQC complete
```

Let's have a more detailed look at the results.

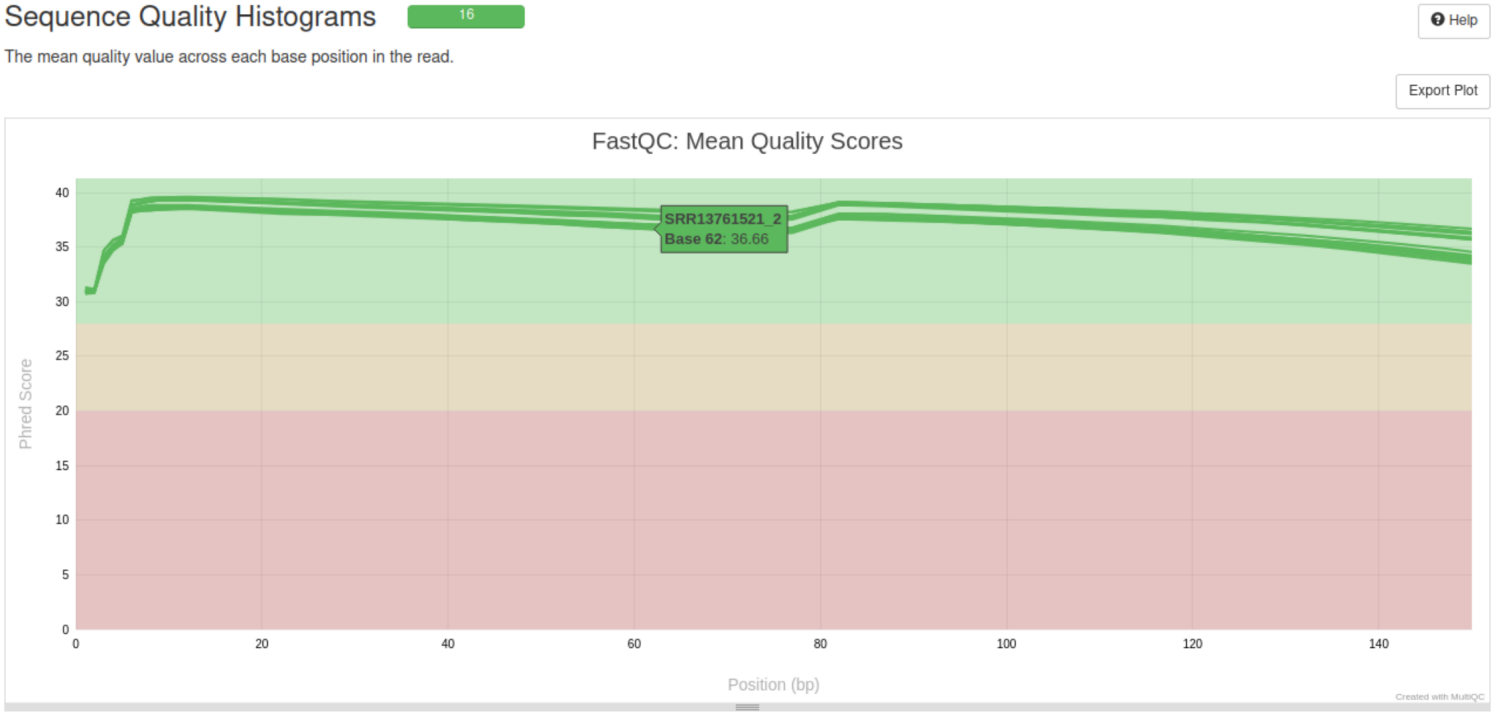
General Statistics

Copy table | Configure columns | Scatter plot | Violin plot | Showing 16/16 rows and 3/6 columns. | Export as CSV

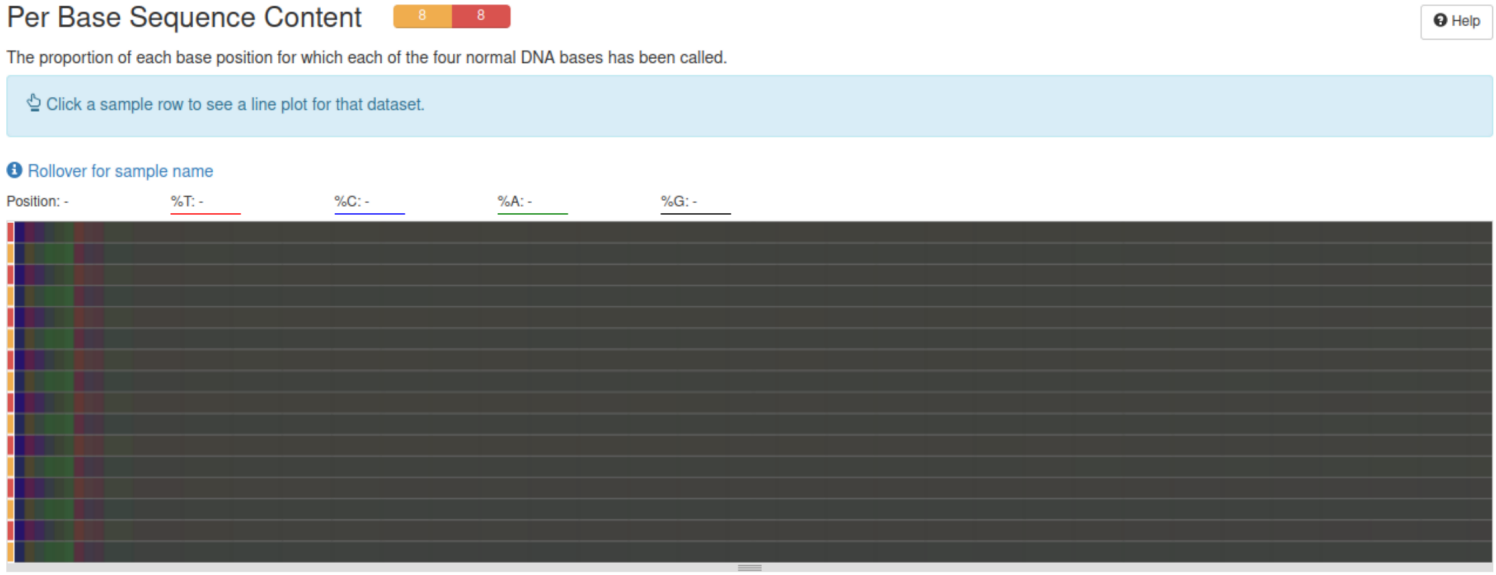
Sample Name	% Dups	% GC	M Seqs
SRR13761520_1	33.6%	48%	21.2M
SRR13761520_2	28.7%	48%	21.2M
SRR13761521_1	33.5%	48%	21.6M
SRR13761521_2	29.6%	49%	21.6M
SRR13761522_1	34.4%	48%	21.1M
SRR13761522_2	29.8%	49%	21.1M
SRR13761523_1	32.8%	48%	19.3M
SRR13761523_2	28.5%	49%	19.3M
SRR13761524_1	33.4%	48%	20.4M
SRR13761524_2	28.3%	49%	20.4M
SRR13761525_1	33.7%	48%	21.4M
SRR13761525_2	28.3%	48%	21.4M
SRR13761526_1	34.4%	48%	20.1M
SRR13761526_2	32.2%	49%	20.1M
SRR13761527_1	35.6%	48%	20.7M
SRR13761527_2	33.6%	49%	20.7M

So far, so good. Each of the files contains around 21 million reads, with a GC content that's normal for the mouse ([source?]). The amount of sequences flagged as duplicates is around 30%, which is to be expected due to the amplification steps during the library preparation ([quote from Ioana Lemnian]).

The Phred scores are, on average, also good across the entire length of the reads., even though, as usual, the quality at the beginning, in the first 10 nucleotides or so from the 5' end, is a bit lower than in the rest of the sequence.



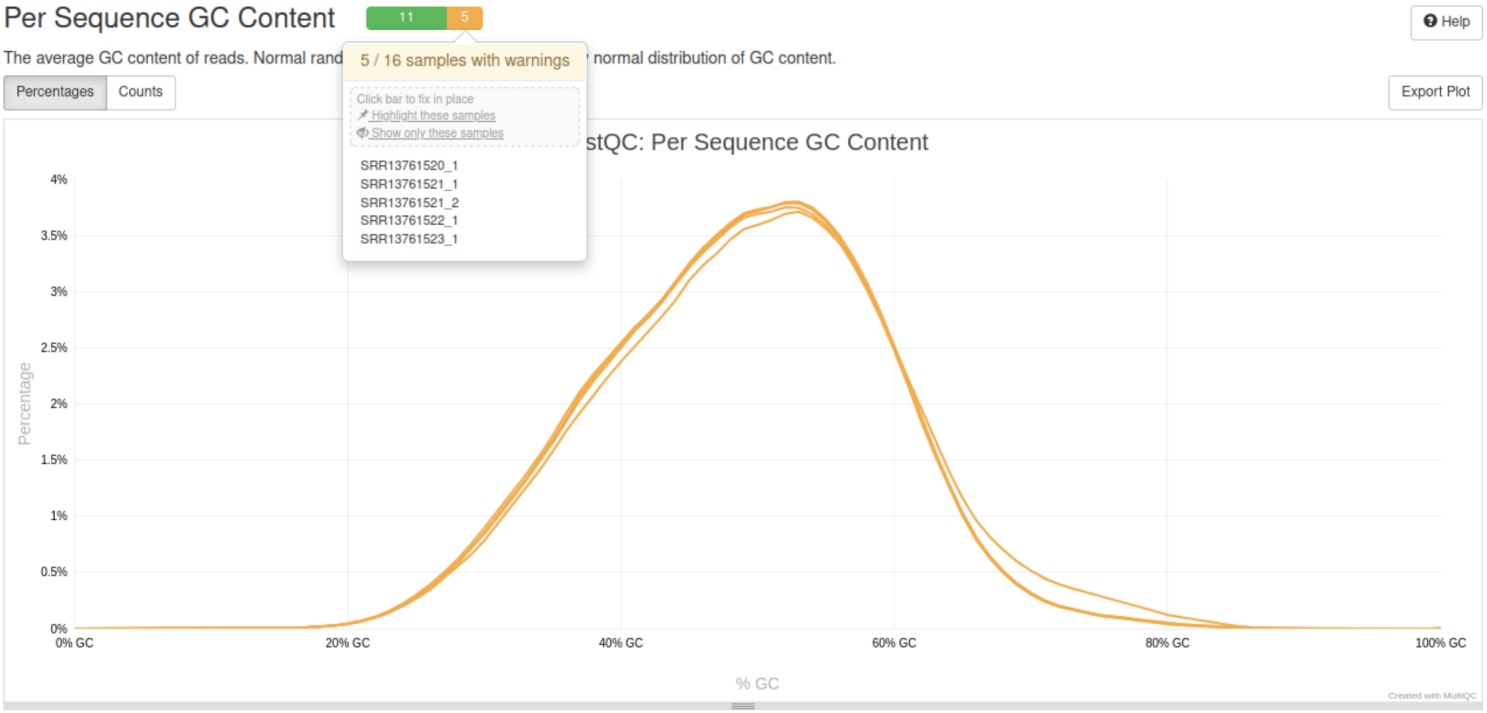
What looked a bit less rosy was the per base sequence content quantification. For all of the `_2` ending samples, FastQC gave a warning for the nucleotide composition, which should ideally be uniform across the read, with each base keeping its proportion percentage and giving a plot with four parallel lines. It's normal that the beginning of the read is a bit more noisy, up to 10 bases, and that's what we see here. Even for the samples ending in `_1` , that were flagged as "failed", the irregularities are restricted to this area. This looks like a good case for trimming these fragment start regions off in subsequent steps.



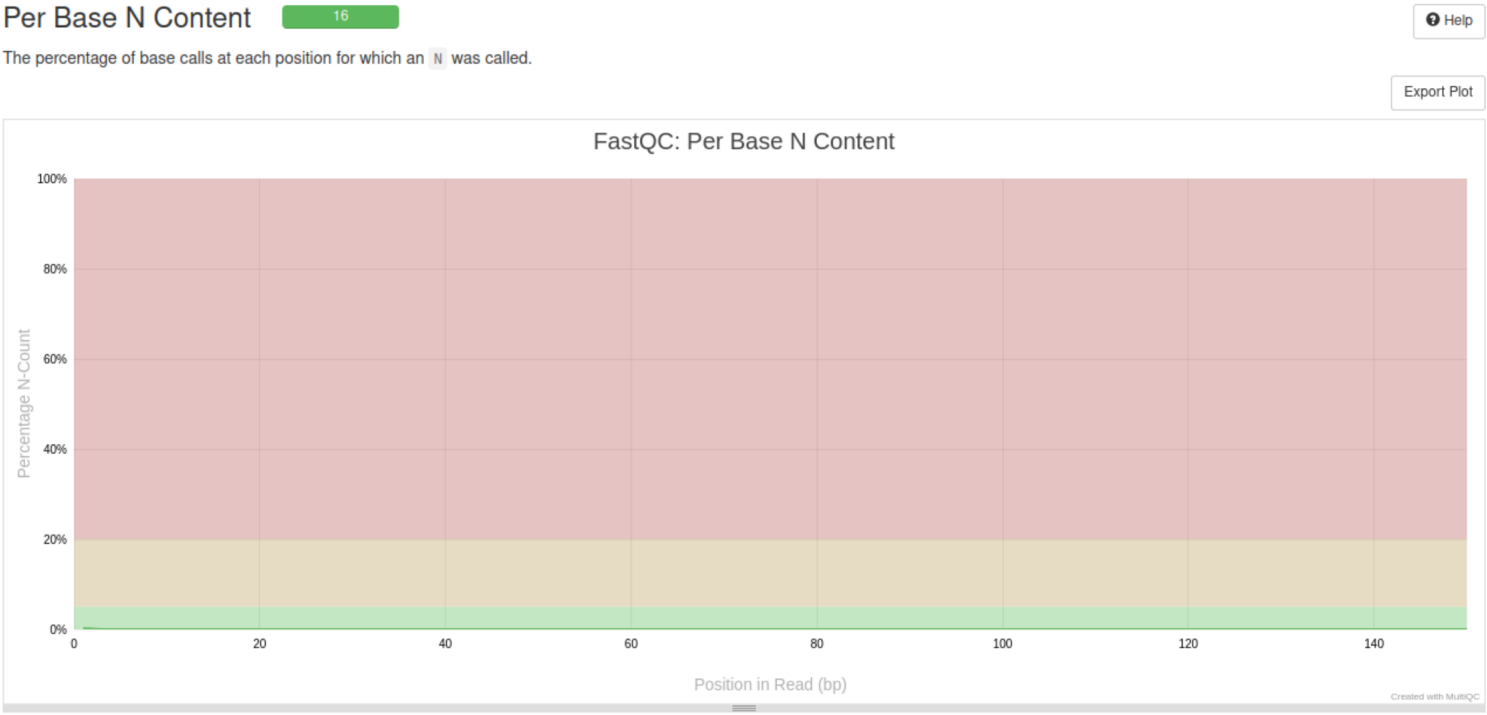
The per sequence GC content looked mostly alright (the reads in 11 of the 16 FastQ files passed with no warning), but there were some where FastQC gave a warning, so I know to keep an eye out in case these files cause some strange, systematic error down the line.

SRR13761520_1
SRR13761521_1
SRR13761521_2
SRR13761522_1

SRR13761523_1



The content of bases that could not be unequivocally identified during the sequencing was, fortunately, negligible.



Many of the FastQ files were found to contain reads from duplications. These usually result from the PCRs from the library preparation or even during sequencing itself, but the concentration of the second peak at 10x duplication suggests that these all stem from some 10-cycle-PCR. FastQC seems to flag any FastQ files where more than 8% of the reads are duplicates at the >10x mark but only those where the dupes surpass 11% of the read count for the 2x duplication mark [why?].

There were no warnings for overrepresented sequences, and FastQC did not find any of the usual sequences of adapters used in NGS kits in the FastQ files. This tells me that the files were processed before their further use, perhaps directly by the sequencing facility after demultiplexing [maybe ***bcl2fastq*** or ***BCL convert*** can also do this? Or maybe the authors just uploaded the already trimmed reads to GEO.]

Summary of initial quality warnings for the reads

Datasets with warnings for GC content:
SRR13761520_1
SRR13761521_1
SRR13761521_2
SRR13761522_1
SRR13761523_1

Dupe warnings:
SRR13761520_1
SRR13761521_1
SRR13761522_1
SRR13761523_1
SRR13761524_1
SRR13761525_1
SRR13761526_1
SRR13761526_2
SRR13761527_1
SRR13761527_2

Trimming the reads

After MultiQC aggregated the FastQC results, I wanted to look specifically if anything needed trimming. I installed Trimmomatic from conda with `conda install bioconda::trimmomatic`

Then, I wrote a tiny script to run Trimmomatic automatically over all sequences and then have FastQC go over the now-trimmed sequences, plus MultiQC to aggregate the results:
[adapter trimming? check MultiQC!]

`!#/bin/bash

`trimmomatic PE -phred33 *.fastq -baseout "output" LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:3

`fastqc output_* --threads 8 --memory 40000 --outdir FastQC_results

`multiqc FastQC_results/ --outdir MultiQC_results/`

I found out in another analysis that one can create at most 4 threads on an octo-core PC like mine[source with explanation], and that trimmomatic has some restriction that only allows the use of a max of 10 GB of RAM, and not 40, as I had indicated to it here, so I adjusted the parameters for this case. [can max RAM be overridden with some option of trimmomatic?]

> **Title**
>
> Contents
> Since the reads all had issues up to around the 10th base from the 5' end, I set the

Dupe warnings:
SRR13761520_1
SRR13761521_1
SRR13761522_1
SRR13761523_1
SRR13761524_1
SRR13761525_1
SRR13761526_1
SRR13761526_2
SRR13761527_1
SRR13761527_2

Downloading genomes

I knew the next step would be to map the reads to the source genome, and for that I, of course, needed the mouse genome.

I found here that NCBI has two command line tools for Linux that ease the download of datasets. From within the NGS environment and in a new folder "Genomes", I used `curl` to get the installers as shown on the NCBI page, and then made the binaries executable as instructed.

I then proceeded to download the genomes that are relevant to me (mouse and human) using the option to find them by accession number. I got the accessions for the latest assemblies here and here. To be on the safe side, I decided to use the more elaborately curated RefSeq assemblies, so the commands looked as follows:

```
(NGS) iweber@iweber-VirtualBox:~/Documents/Genomes$ datasets download genome accession GCF_000001405.40 --filename genome_H_sapiens_GRCh38.zip
```

...and got a big, fat load of nothing xD, because the shell said it can't find the command "datasets". Instead of fumbling to find where the binaries associated with these two tools are, I decided to simply use conda to install them directly into this environment.

`conda create -n ncbi_datasets` conda activate ncbi_datasets conda install -c conda-forge ncbi-datasets-cli`

```
(NGS) iweber@iweber-VirtualBox:~/Documents/Genomes$ conda activate ncbi_datasets
(ncbi_datasets) iweber@iweber-VirtualBox:~/Documents/Genomes$ conda install -c conda-forge ncbi-datasets-cli
Channels:
 - conda-forge
 - bioconda
 - defaults
Platform: linux-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/iweber/anaconda3/envs/ncbi_datasets

  added / updated specs:
    - ncbi-datasets-cli


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    ncbi-datasets-cli-16.14.0  |       ha770c72_2        15.8 MB  conda-forge
    ------------------------------------------------------------
                                           Total:        15.8 MB

The following NEW packages will be INSTALLED:

  ca-certificates    conda-forge/linux-64::ca-certificates-2024.2.2-hbcca054_0
  ncbi-datasets-cli  conda-forge/linux-64::ncbi-datasets-cli-16.14.0-ha770c72_2


Proceed ([y]/n)? y


Downloading and Extracting Packages:

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
(ncbi_datasets) iweber@iweber-VirtualBox:~/Documents/Genomes$
```

Indexing the mouse genome for RNA-seq

Mapping

Installing STAR

`conda install bioconda::star`
`conda install bioconda::subread`

R, RStudio

In R:
`if (!require("BiocManager", quietly = TRUE)) install.packages("BiocManager") BiocManager::install("DESeq2")`

Installing R and RStudio on the virtual machine

From https://cran.r-project.org/

```
# update indices
sudo apt update -qq
# install two helper packages we need
sudo apt install --no-install-recommends software-properties-common dirmngr
# add the signing key (by Michael Rutter) for these repos
# To verify key, run gpg --show-keys /etc/apt/trusted.gpg.d/cran_ubuntu_key.asc
# Fingerprint: E298A3A825C0D65DFD57CBB651716619E084DAB9
wget -qO- https://cloud.r-project.org/bin/linux/ubuntu/marutter_pubkey.asc | sudo tee -a
/etc/apt/trusted.gpg.d/cran_ubuntu_key.asc
# add the R 4.0 repo from CRAN -- adjust 'focal' to 'groovy' or 'bionic' as needed
sudo add-apt-repository "deb https://cloud.r-project.org/bin/linux/ubuntu $(lsb_release -cs)-cran40/"
```

Install visual studio code

I downloaded the Ubuntu installer from https://code.visualstudio.com/download I opened it with an app but it did not show up as an app afterwards. So, looked for it in the snap store and found the app itself, called Code.

[]img code in snap

...but that gave me an error message.

However, after restarting the installation, it went to 100% and now it appears as an app, which I can open and use.