High-Performance Computing & Biology

Paul Johnston

paul.johnston@fu-berlin.de

Berlin Center for Genomics in Biodiversity Research | Leibniz-Institut für Gewässerökologie und Binnenfischerei | Freie Universität Berlin

December 2021

I might use terms interchangeably but:

- a computing cluster is a type of HPC
- bioinformatics is type of data science

sorry if I confuse anyone

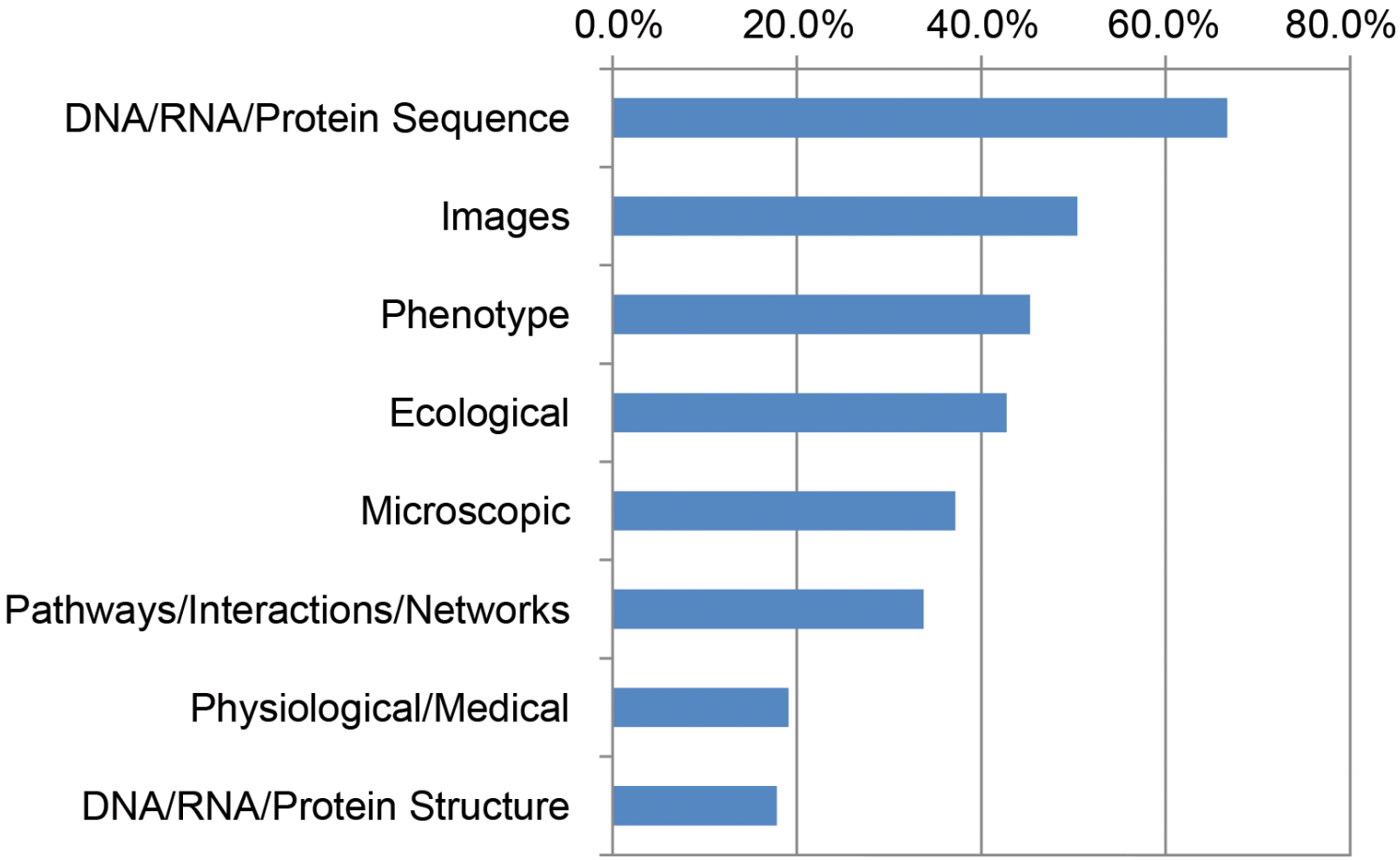
94% of students/faculty/researchers use large data sets or will in the near future (n = 1,097)

47% rated their bioinformatics skill level as “beginner,” (n = 608)

58% felt their institutions do not provide all the computational resources needed for their research (n = 1,024)

https://doi.org/10.1371/journal.pcbi.1005755

Biologists receive little training in data science...

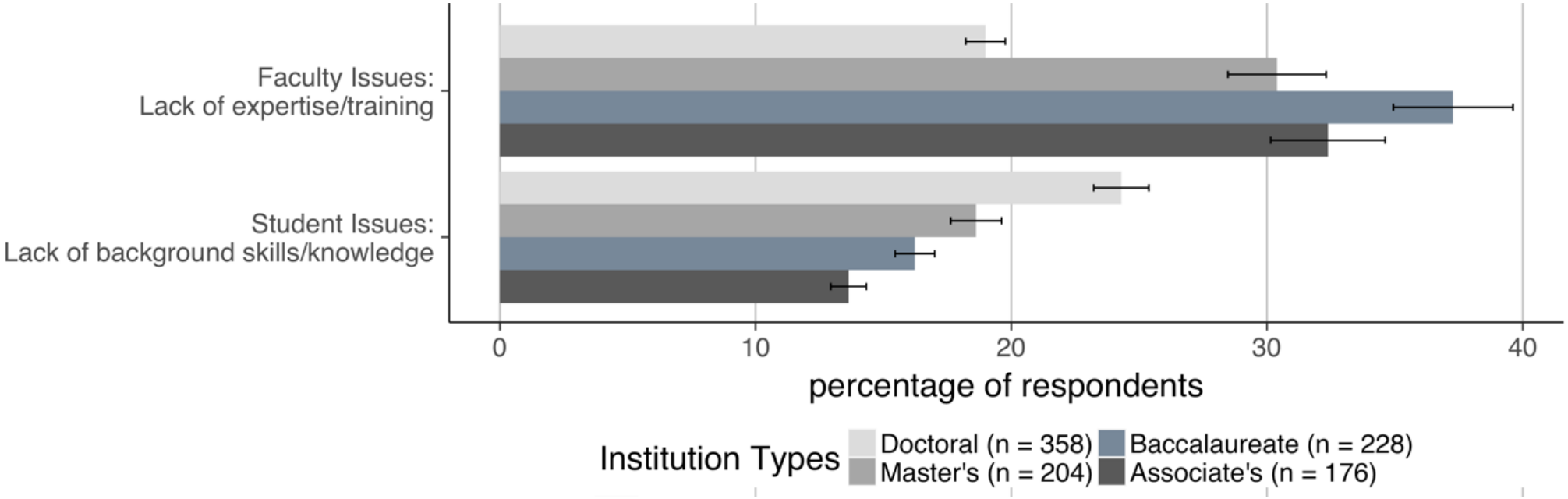


but most projects require it..

https://doi.org/10.1371/journal.pcbi.1005755

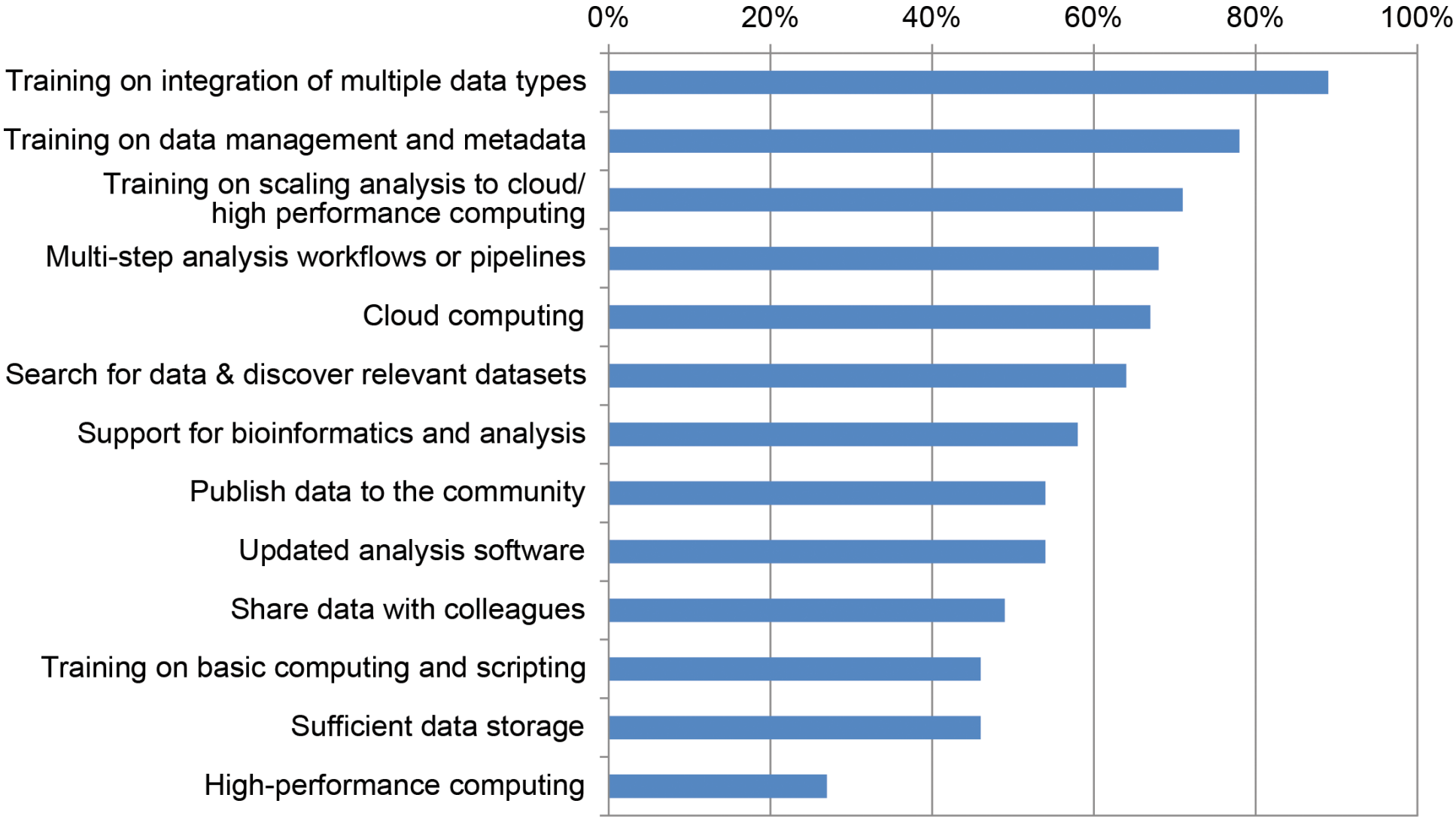
Barriers to teaching bioinformatics

Decade of Highest Degree Earned	Formal Bioinformatics Training (%)	Faculty Integrating Bioinformatics (%)
1980–1989	8.4	35.4
1990–1999	11.3	41.9
2000–2009	35.1	41.7
2010–2016	48.3	25.2



https://doi.org/10.1371/journal.pone.0224288

Unmet needs in bioinformatics



https://doi.org/10.1371/journal.pcbi.1005755

“These studies suggest a scenario of big data inundating unprepared biologists.”

https://doi.org/10.1371/journal.pcbi.1005755

(d) *De novo* assembly and annotation

Assemblies for both species were produced using Trinity v. 2.8.4 [29], incorporating quality and adapter filtering via Trimmomatic [30] and subsequent *in silico* normalization. Assemblies were annotated with the Trinotate annotation pipeline [31].

(e) Inference of orthologous gene groups

OrthoFinder v. 2.2.7 [32] was used to infer orthology between predicted peptide sequences from *Gr. bimaculatus*, *G. mellonella* and the proteomes of other insect species with sequenced genomes. These included *Acyrthosiphon pisum*, *Apis mellifera*, *Drosophila melanogaster*, *Tribolium castaneum*, *Zootermopsis nevadensis*, as well as all Lepidopteran genomes from lepbase release 4 [33].

(f) Immune effector gene identification

Immune effectors of *Gr. bimaculatus* and *G. mellonella* were identified from orthologue groups containing annotated immune genes from previously published insect genome projects. Additionally, blast and HMM homology searches were performed using previously described insect immune effector proteins as queries against each *de novo* assembly.

(g) Differential gene expression

For both species, transcript abundances were quantified by pseudo-aligning RNAseq reads to *de novo* assemblies using Salmon v. 0.1.2.0 [34]. tximport [35] was used in conjunction with DESeq2 [36] to model gene-level estimated counts while correcting for changes in transcript usage across samples. Specifically, to identify differential expression as a function of developmental stage, likelihood-ratio tests were performed between full and intercept-only negative binomial GLMs. Differential expression was considered to be significant when fold changes were greater than 2 for pairwise Wald contrasts of developmental stages, with a false discovery rate (FDR)-corrected *p*-value of less than 0.05. The mean of the normalized counts for each gene was used as the informative covariate for indepen-

- The fine details of the computation aren't reported (memory, storage, time etc)
- it's easy for others to underestimate the required resources
- computing skill set is also unclear

Storage problems

RNAseq of 35 samples

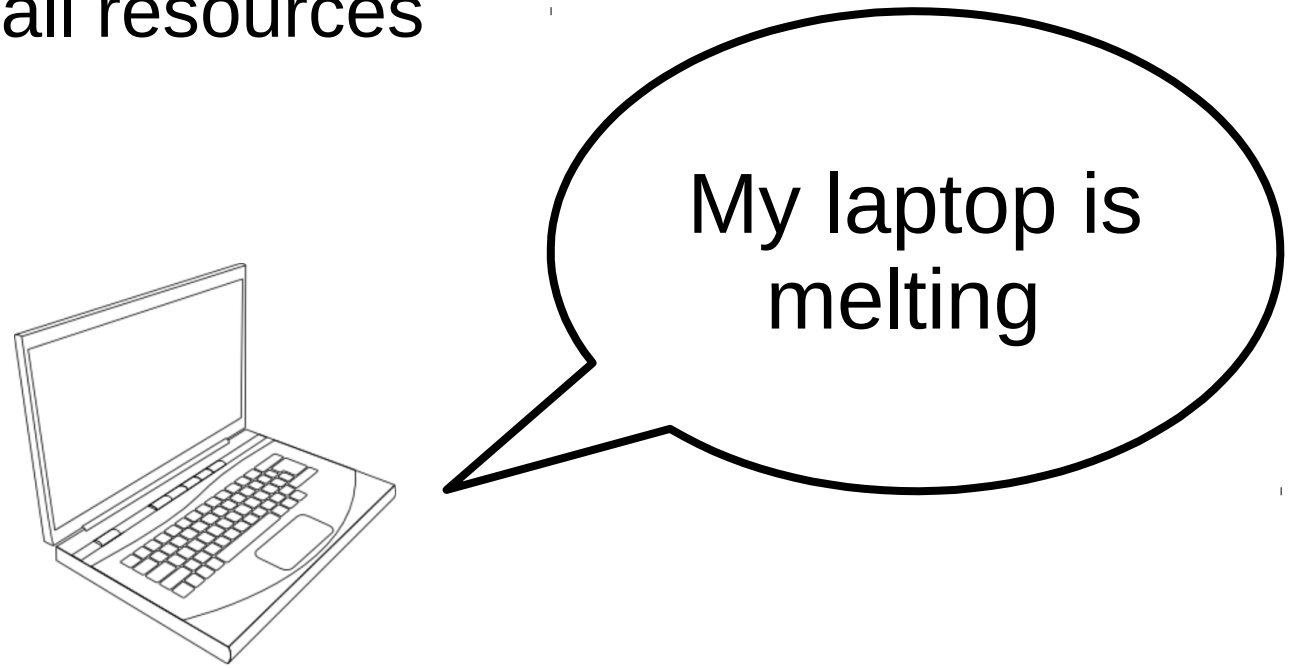
- one lane of NovaSeq
- 35 compressed fastq files
- >400 GB (compressed)

The storage is full before I've even started!

Time problems

alignment of 35 RNAseq samples to genome

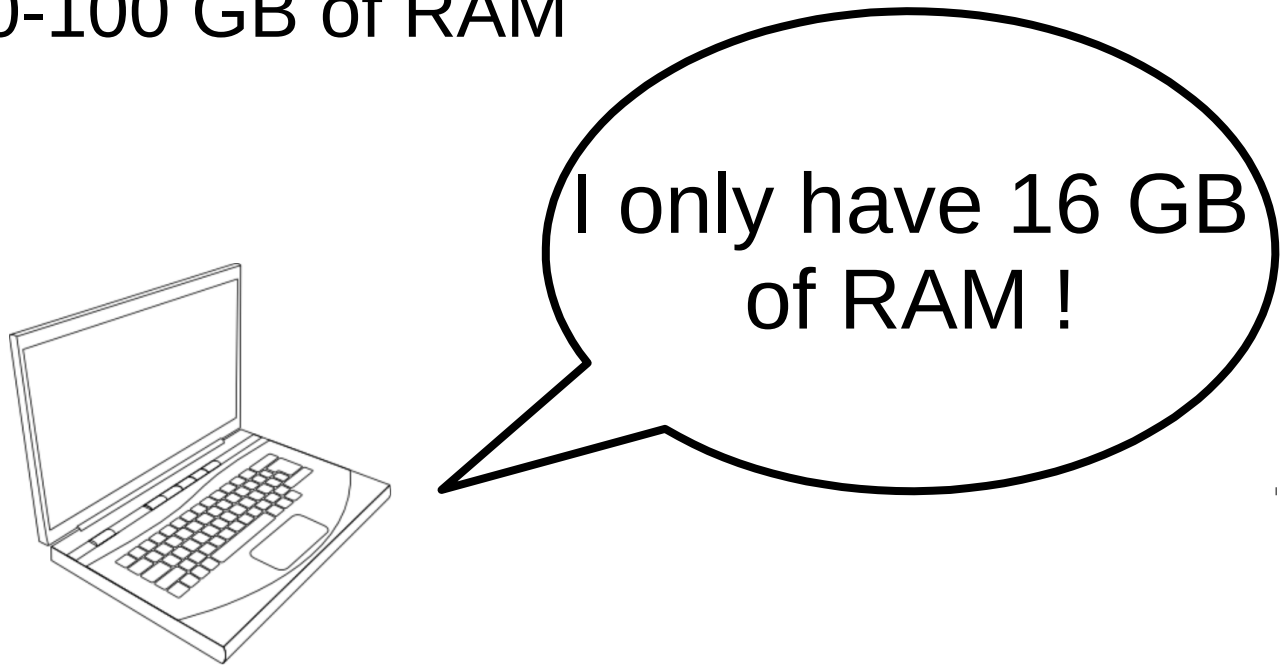
- Each alignment takes 5 hours
- Will run for 7 days
- Will use all resources



Memory problems

Quantification of 35 RNAseq samples gives

- a matrix of 10,000s of counts x 35
- need to fit 10,000s of models
- needs 20-100 GB of RAM

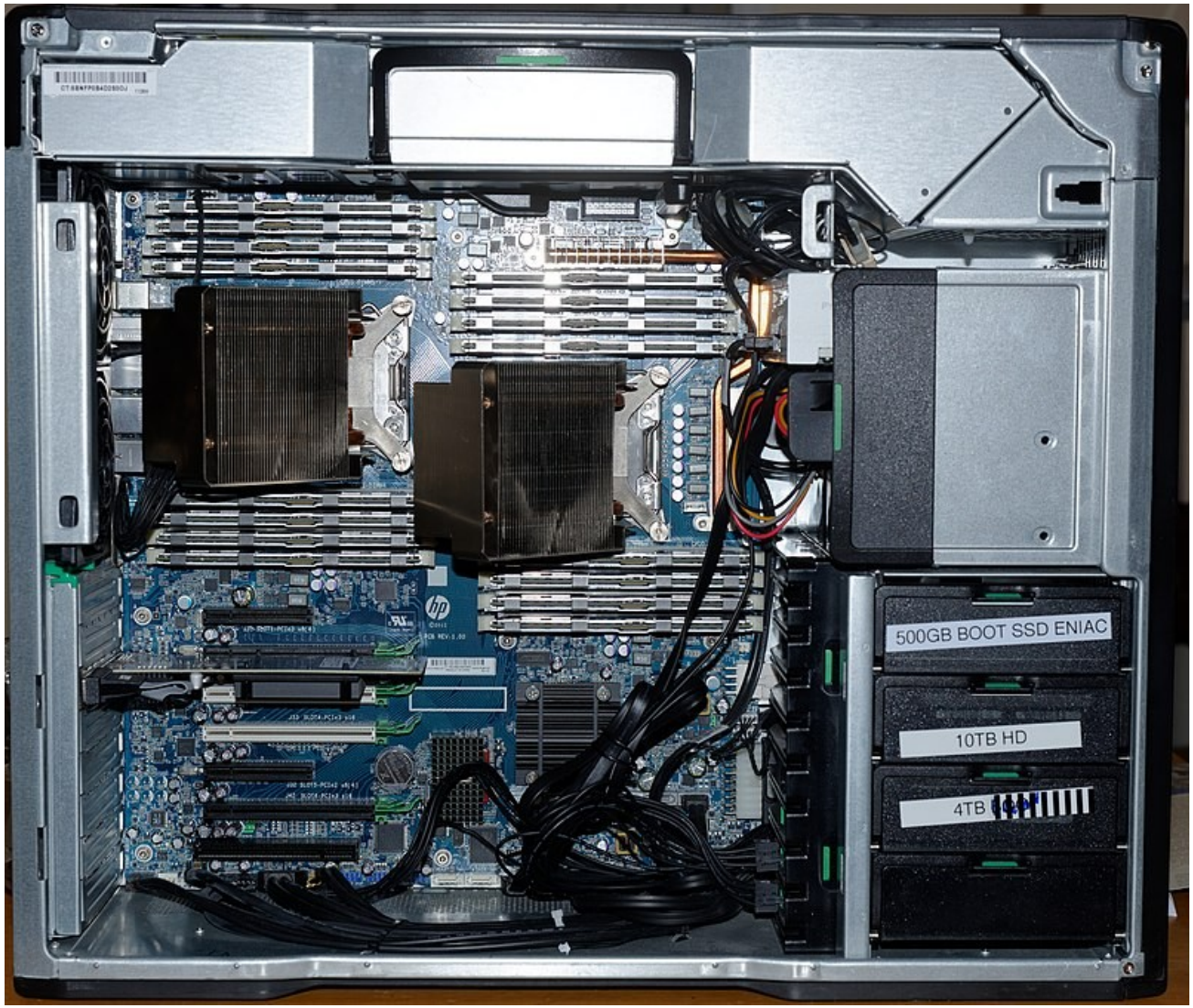


Lots of problems

de novo assembly of 35 RNAseq samples

- needs >100 GB RAM, runs for many days
- needs many CPUs, generates TBs of files
- repeat the assembly many times with different parameters

Workstation (1,000s €)



Server (10,000s €)

- e.g. 36,000€
  - 1.5 TB RAM
  - 40 cores
  - 48 TB storage
- Plus energy, cooling, rack etc.

Computing cluster (HPC)
100,000s €

Important points

Every university has (access to) a cluster

Usage is generally free or extremely cheap

There are university employees to provide support

The system is highly robust and stable

i.e. it’s there and it works now

List of results

Projects | People | Institutions

Your Projects query is

Keyword(s): **boris proppe**

- **Include projects without final report**

start new search | change search

More Results

Based on your query, more matches were found in the following areas

→ People (1)
→ Institutions (1)

results per page: 5 | **10** | 25 | 50

Order by: **Relevance** | Name | GZ

« ‹ Results **1 to 2 out of 2** on **1 page** › »

High-Performance-Computing-Cluster und Speichersystem

Leader	**Boris Proppe**		
DFG Programme	**Major Research Instrumentation**	Term	**in 2010**

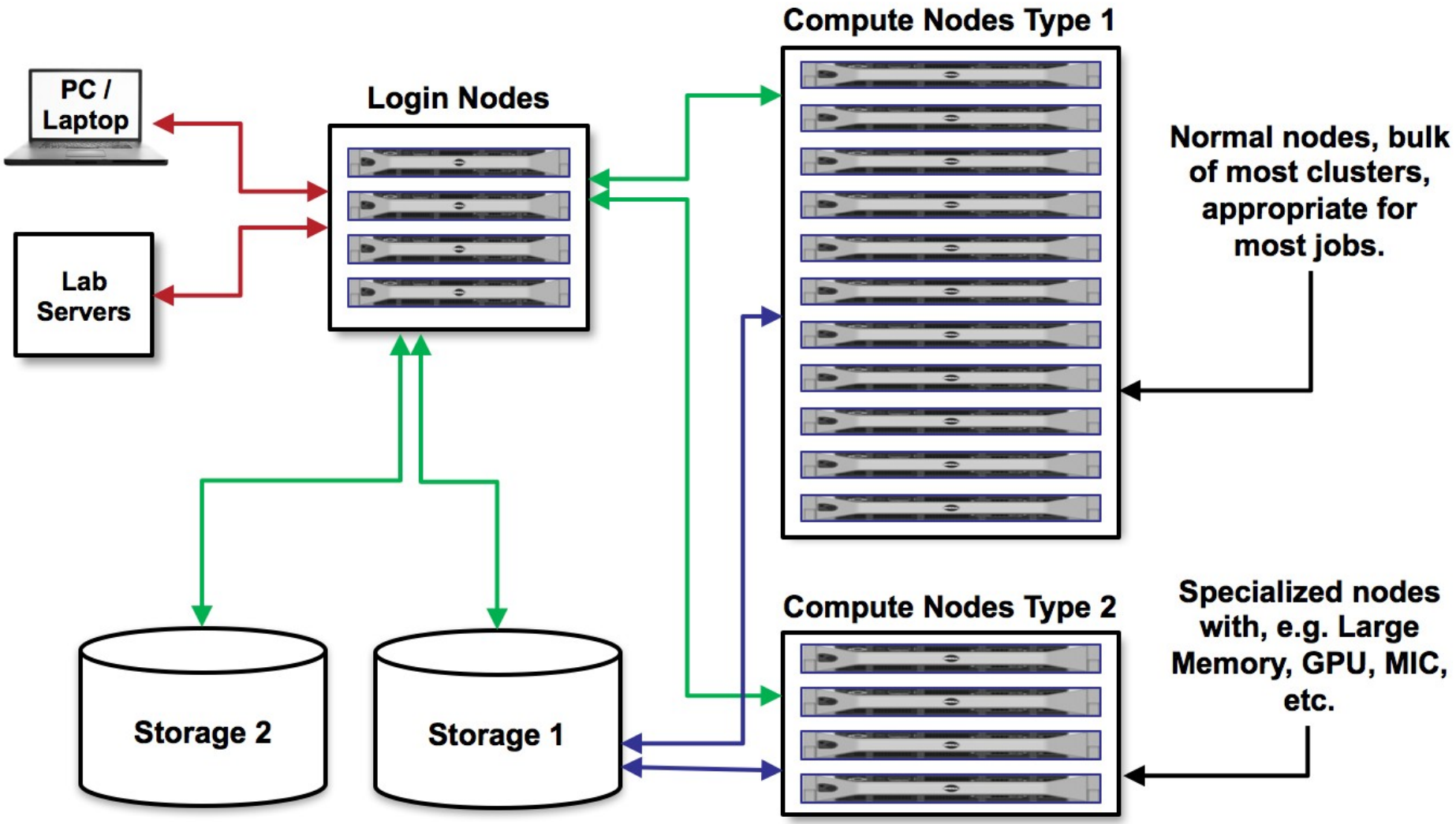
High-Performance-Computing-Cluster und Speichersystem

Leader	**Boris Proppe**		
DFG Programme	**Major Research Instrumentation**	Term	**in 2017**

Das Hochschulrechenzentrum (ZEDAT) und die IT-Dienste der Fachbereiche Mathematik/Informatik und Physik der Freien Universität Berlin beantragen gemeinsam die ...

→ more

Typical cluster



Terminology

Job: reservation to run commands

Node: physical machine, part of cluster

Core/CPU: processing unit, nodes contain many CPUs

Partition: nodes may be organized into partition
e.g. begendiv bought large-memory nodes

Curta specifications

Specifications

Nodes	170x Intel CPUs only, 12x Asus with GPUs
Processors	2x Xeon Skylake 6130 (2.1 GHz, 16 cores, 22 MB Cache) per node
GPUs	24x NVIDIA GTX1080Ti
RAM	3/6/12/24 GB per core (96/192/384/768 GB per node) 103/50/8/4 nodes
File System	1.8 PB /scratch with Lustre
Network	10 Gigabit-Ethernet - TCP/IP
Interconnect	Omnipath
Operating System	CentOS 7
Batch-System	Slurm

The DOI 10.17169/refubium-26754 refers to a more complete description of the system and should be refered to in any publications.

Connecting to the cluster

```
# remember to use your zedat username
ssh USERNAME@curta.zedat.fu-berlin.de
```

Connecting to the cluster

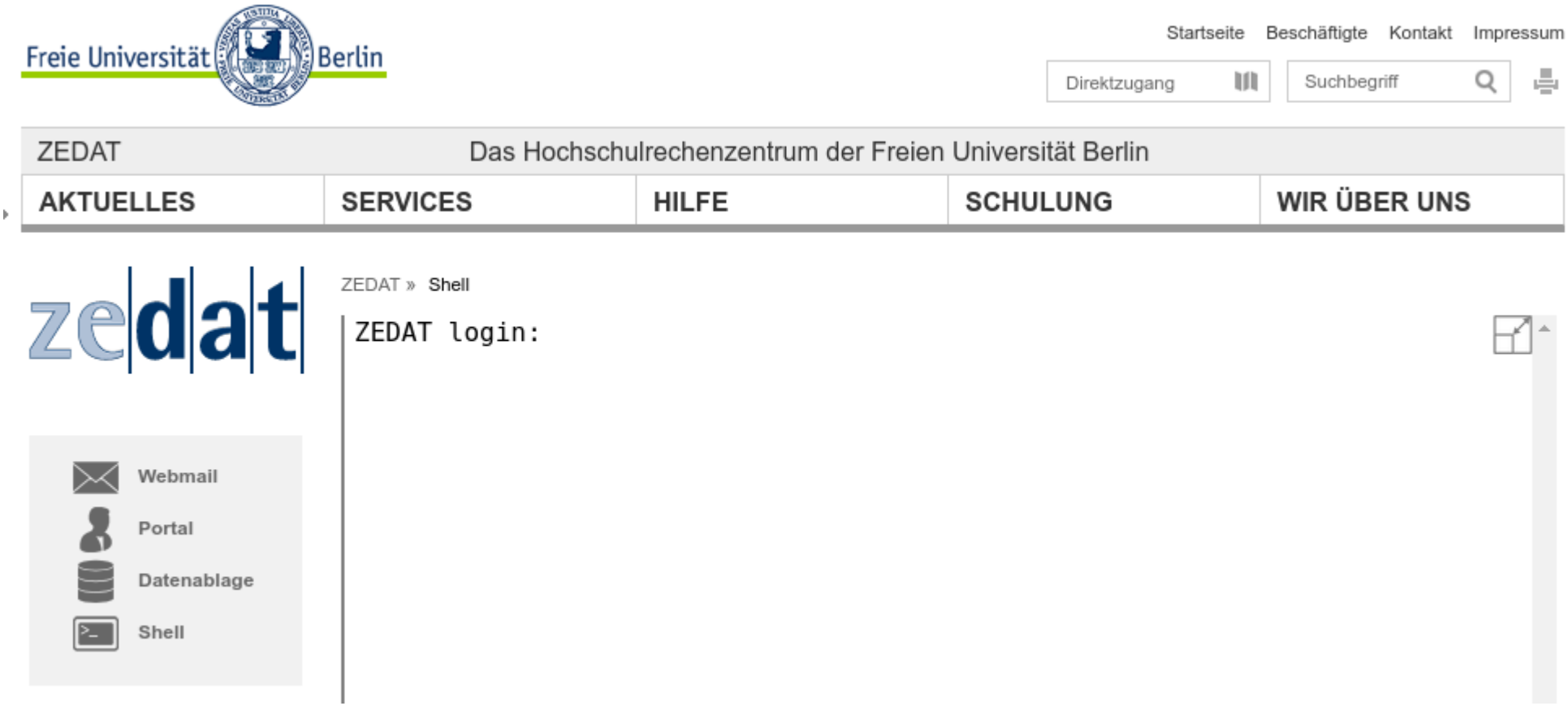
```
# remember to use your zedat username
ssh USERNAME@curta.zedat.fu-berlin.de
```

- the FU VPN needs to be active (at home)
- if the VPN doesn’t work on your system then you can first connect to the zedat login server

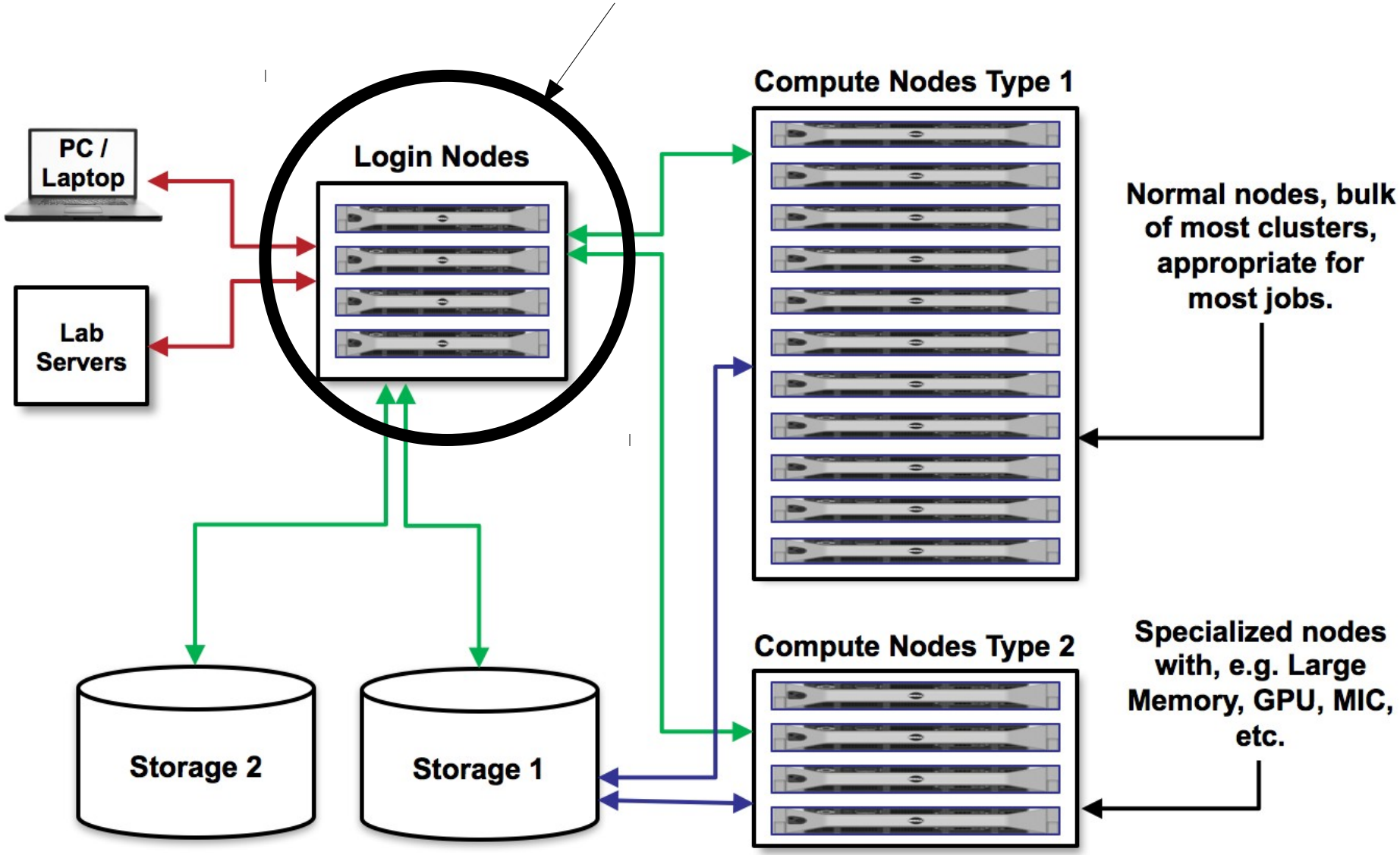
```
# connect to login server
ssh USERNAME@login.zedat.fu-berlin.de
# you are now connected to the login server
# from here you can connect to the cluster
ssh USERNAME@curta.zedat.fu-berlin.de
```

Connecting to the cluster

- zedat also provides a shell here:
  https://www.zedat.fu-berlin.de/Shell



You are now on one of these nodes



Login nodes

These are for basic tasks:

- Uploading data # scp, rsync, wget, etc
- Managing files # cp, mv, gunzip
- Compiling software # configure, make
- Editing scripts # nano, vim
- Checking/managing jobs # squeue

Important note

The login nodes are for setting things up and submitting jobs to the compute nodes

Running commands on the login nodes is bad

- You would get an annoyed email
- It slows/crashes the node for other users

Compute nodes

Job scripts are sent here to run

- resources allocated by workload manager (Slurm)

- other workload managers/schedulers exist
    e.g. PBS, SGE, LFS

- resources are allocated according to:
    system load
    fair share algorithms

Basic conventions

```
# remember to use your zedat username
ssh USERNAME@curta.zedat.fu-berlin.de
```

/home/$USER/
- limited space
- backed up nightly
- store software, config files

/scratch/$USER/
- lots of storage space
- no backup
- default working space

Different cultures exist in different clusters
e.g. precise location of scratch, backup policy

First job on the cluster

```
# remember to use your zedat username
ssh USERNAME@curta.zedat.fu-berlin.de
# enter the following command
echo “Hello World”
# “Hello World” should print to your screen (the standard output)
```

First job on the cluster

```
# remember to use your zedat username
ssh USERNAME@curta.zedat.fu-berlin.de
# enter the following command
echo “Hello World”
# “Hello World” should print to your screen (the standard output)
```

We just ran a command on the login node

(this is OK because it is not intensive)

Now lets submit it as a job to a compute node

First job on the cluster

```
# we should work in our scratch directory
# so navigate to your scratch directory
cd /scratch/$USER/
# now make a directory with a sensible name to work in
# yearmonthday_sensible_name is a good naming convention
mkdir 20211215_lvdatascieeb
# navigate inside the new directory
cd 20211215_lvdatascieeb
# copy the example job script from my scratch directory to yours
cp /scratch/perugolate/20211215_lvdatascieeb/20211215_hello_world.sh .
# have a look at the script
cat 20211215_hello_world.sh
```

Example job script

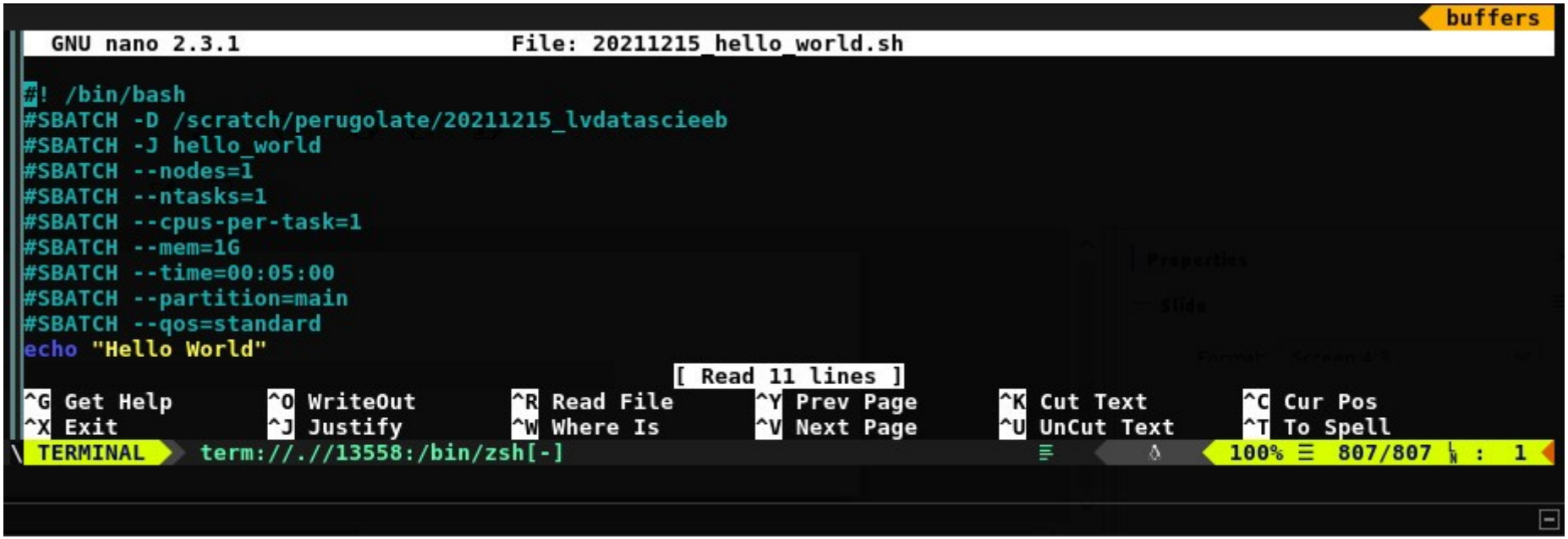
```
#! /bin/bash
#SBATCH -D /scratch/USERNAME/20211215_lvdatascieeb      # directory
#SBATCH -J hello_world          # name of job
#SBATCH --nodes=1               # requested nodes
#SBATCH --ntasks=1              # number of tasks in job
#SBATCH --cpus-per-task=1       # requested CPUs
#SBATCH --mem=1G                # requested RAM
#SBATCH --time=00:05:00         # requested time
#SBATCH --partition=main        # partition where job will run
#SBATCH --qos=standard          # complicated
echo "Hello World"
```

- line 2 specifies the directory where the job will run

- we need to edit it

Example job script

open the job script in the text editor ‘nano’
nano 20211215_hello_world.sh
navigate with arrow keys to USERNAME on line 2
delete and enter your username
when finished editing, press control+X to exit
then press shift+Y to save changes
(or maybe shift+J if your language is set to german, I’m not sure)



Submit the job

```
# submit the job script using
sbatch 20211215_hello_world.sh
```

sbatch is a command from the slurm workload/schedule managing software

squeue is used to monitor the queue (or squeue --me)

```
squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
           8695474  agkeller     KF-2   heif89  R 1-06:23:29      1 g014
           8694500  agkeller    LiI-3   heif89  R 1-06:50:38      1 g013
           8717240  begendiv jamesTes  james94  R    3:07:30      1 b004
           8717994  begendiv GEP.merq ddepanis  R       6:30      1 b002
           8717993  begendiv GEP.merq ddepanis  R      34:32      1 b003
           8717992  begendiv GEP.merq ddepanis  R      37:32      1 b004
           8717991  begendiv GEP.merq ddepanis  R      46:58      1 b001
           8717989  begendiv GEP.busc ddepanis  R      53:25      1 b002
           8717983  begendiv GEP.busc ddepanis  R    1:36:20      1 b003
```

Submit the job

```
# submit the job script using
sbatch 20211215_hello_world.sh
```

The job should complete instantly

You will notice “Hello World” was not printed to your screen

The standard output was redirected to a file slurm-[jobID].out

Job output

```
# slurm has also added some usage stats to the end of the file
cat slurm-8722119.out
Hello World

== Epilog Slurmctld
==========================================================

Job ID: 8722119
Cluster: curta
User/Group: perugolate/agrolff
State: COMPLETED (exit code 0)
Cores: 1
CPU Utilized: 00:00:00
CPU Efficiency: 0.00% of 00:00:01 core-walltime
Job Wall-clock time: 00:00:01
Memory Utilized: 0.00 MB (estimated maximum)
Memory Efficiency: 0.00% of 1.00 GB (1.00 GB/node)

==========================================================
==============
```

Software

The cluster works just like a linux computer

- you can download/compile software

- you can install a package manager e.g. conda

- the HPC staff can help/advise on installation

- common software is available via “module load”
    multiple versions of python/perl/R
    bioinformatics software

Second job

Let’s adapt one of your R exercises to the cluster

We need to:

- upload your R script and data

- make some edits to the R script

- create a job script to run the R script

Upload you script/data

transfer your R script / data to the the cluster

open a new terminal/shell on your computer and run:

```
# the command should be a single line
scp Exercises_Statistics2_Solutions.R username@curta.zedat.fu-berlin.de:/scratch/USERNAME/20211215_lvdatascieeb
# and the data
scp data_files_08.12.21.zip username@curta.zedat.fu-berlin.de:/scratch/USERNAME/20211215_lvdatascieeb
```

If you don’t have your R script / data from last week, then copy them from my scratch directory:

```
# copy script from my scratch
cp /scratch/perugolate/Exercises_Statistics2_Solutions.R ./
# copy data from my scratch
cp /scratch/perugolate/data_files_08.12.21.zip ./
```

Prepare the data/script

```
# unzip the uploaded data
unzip data_files_08.12.21.zip
```

We are going to prepare exercise 1 (the first 62 lines) from Exercises_Statistics2_Solutions.R as a script

the file paths (C:\\Users\\Lynn\\) show the script was likely not prepared on a unix computer. Lets fix the line endings first:

```
# if a file was prepared on windows or Mac (prior to OS X) the line endings
# will be (invisibly) different, which can cause all sorts of problems
dos2unix Exercises_Statistics2_Solutions.R
```

```
# copy first 62 lines to a new file with a sensible name
head -n 62 Exercises_Statistics2_Solutions.R > 20211215_exercise_01.R
```

Modify script

```
# open the script in nano to edit
nano 20211215_exercise_01.R
```

We are not using windows so change line 3 from:

```
filestem <- 'C:\\Users\\Lynn\\'
```

to:

```
#filestem <- 'C:\\Users\\Lynn\\'
```

the hash character ensures that this line is ignored

Next we modify line 9 to read:

```
dat <- read.table("LG_Data_Phytoplankton_Species1.txt", header=T)
```

Save the changes

Make a job script

We can reuse the sbatch parameters from the first job script

Lets copy the first 10 lines to a new script file:

```
# copy the sbatch parameter lines from first job script
head -n10 20211215_hello_world.sh > 20211215_exercise_01.sh
# open the new file in nano for editing
nano  20211215_exercise_01.sh
```

Make a job script

I have made the following edits in nano

- changed job name on line 3 from “hello_world” to “ex_1”
- added 2 commands to the bottom (in bold)

```
#! /bin/bash
#SBATCH -D /scratch/perugolate/20211215_lvdatascieeb
#SBATCH -J ex_1
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=1
#SBATCH --mem=1G
#SBATCH --time=00:05:00
#SBATCH --partition=main
#SBATCH –qos=standard
module load R/4.1.0-foss-2021a
Rscript 20211215_exercise_01.R
```

Make a job script

Many versions of R are already installed on the cluster

It is also possible to
- download and compile it yourself
- install it using a package manager such as conda

Here, we have loaded R using:

```
module load R/4.1.0-foss-2021a
```

This is a safe option because it has been compiled properly by the HPC staff

You can see the other software available by using:

```
module avail
```

Submit the job script

```
module load R/4.1.0-foss-2021a
```

The job should complete within a few seconds

Again nothing is printed to the screen. This has been redirected to a text file slurm-[jobID].txt

```
head slurm-8726728.out
Call:
lm(formula = Size ~ TP * Trans, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-4.0573 -0.9617  0.0516  0.9243  3.1775

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
```

Job output

R was not run in interactive mode
- i.e. there was no window to display the plots

In this case Rscript redirects the plots to Rplots.pdf

You will need to transfer the file to your computer in order to view it

Open a terminal on your computer and type:

```
# replace USERNAME with your username
scp username@curta.zedat.fu-berlin.de:/scratch/USERNAME/20211215_lvdatascieeb/Rplots.pdf ./
```