# Class 13

Mirte Ciz Marieke Kuijpers

02/03/2022

## Introduction

In this class we learnt how to use the command line and then utilized this knowledge on a remote server to do some computationally heavy blast searches of a large set of proteins from mice against the proteome of Zebrafish.

## Homework Questions

### Q1. [6pt] List the UNIX bash shell commands to:

**• open a secure shell on a remote machine:**

Use `shh username@remotemachineIP` if a key is required add this with the option `-i`. For example, I used `ssh -i "bggn213_mkuijpers.pem" ubuntu@ec2-34-217-214-87.us-west-2.compute.amazonaws.com`. Note that my passkey was in my working directory when I used this command; if it had not been I would have needed to indicate the path to the key.

**• make a new folder in your home area called "test":**

```
mkdir test
```

**• download this file "https://files.rcsb.org/download/5P21.pdb.gz":**

```
wget https://files.rcsb.org/download/5P21.pdb.gz
```

OR

```
curl -O https://files.rcsb.org/download/5P21.pdb.gz
```

**• unzip/decompress it the file:**

```
gunzip 5P21.pdb.gz
```

**• print to screen the first 6 lines:**

```
head -6 5P21.pdb
```

**• print to lines beginning with ATOM to a new file called "coords.pdb":**

```
grep "^ATOM" 5P21.pdb > coords.pdb
```

**Q2. [3pt] List the UNIX commands to copy securely the file "myaln.fa" in your current working directory to your home area on the remote machine "biglabcluster.ucsd.edu":**

The command for copying securly between servers is `scp`. This command has the following syntax `scp [optional parameters such as -i identity_file] source ... target`. Thus the correct code to here would be as follows

```
scp -i /absolute.path.to.passkey/passkey myaln.fa
myusername@biglabcluster.ucsd.edu:~
```

If the passkey is in your current working directory then the path to it does not have to be specified.

**Q3. [1pt] The alignment file "myaln.fa" is not in your current working directory but it is in your "Downloads" directory. Write the R code to import this alignment to the named object "aln" using a function from the bio3d package.**

```
aln <- read.fasta("C:/Users/mirte/Documents/myaln.fa")
```

## Questions posed in the instructions

**Q. How many sequences are in this mouse.1.protein.faa file? Hint: Try using grep to figure this out…**

Use the following code: `grep -c ">" mouse.1.protein.faa` The `-c` option tells grep to only count the number of times it finds the pattern rather than returning each line with the pattern (where the pattern is >). The output is `66946`, so there are 66946 protein sequences in this file.

**Q. What happens if you run the above command without the > mm-first.fa part?**

The > redirects the output away from the terminal/console into a file. Without the `> mm-first.fa` part the output will simply print in the terminal/console. This output is shown below for clarity.

```
>YP_220550.1 NADH dehydrogenase subunit 1 (mitochondrion) [Mus musculus
domesticus]
MFFINILTLLVPILIAMAFLTLVERKILGYMQLRKGPNIVGPYGILQPFADAMKLFMKEPMRPLTTSMSLFIIAPTL
SLT
LALSLWVPLPMPHPLINLNLGILFILATSSLSVYSILWSGWASNSKYSLFGALRAVAQTISYEVTMAIILLSVLLMN
GSY
SLQTLITTQEHMWLLLPAWPMAMMWFISTLAETNRAPFDLTEGESELVSGFNVEYAAGPFALFFMAEYTNIILMNAL
TTI
IFLGPLYYINLPELYSTNFMMEALLLSSTFLWIRASYPRFRYDQLMHLLWKNFLPLTLALCMWHISLPIFTAGVPPY
M
>YP_220551.1 NADH dehydrogenase subunit 2 (mitochondrion) [Mus musculus
domesticus]
MNPITLAIIYFTIFLGPVITMSSTNLMLMWVGLEFSLLAIIPMLINKKNPRSTEAATKYFVTQATASMIILLAIVLN
YKQ
```

```
LGTWMFQQQTNGLILNMTLMALSMKLGLAPFHFWLPEVTQGIPLHMGLILLTWQKIAPLSILIQIYPLLNSTIILML
AIT
SIFMGAWGGLNQTQMRKIMAYSSIAHMGWMLAILPYNPSLTLLNLMIYIILTAPMFMALMLNNSMTINSISLLWNKT
PAM
LTMISLMLLSLGGLPPLTGFLPKWIIITELMKNNCLIMATLMAMMALLNLFFYTRLIYSTSLTMFPTNNNSKMMTHQ
TKT
KPNLMFSTLAIMSTMTLPLAPQLIT
```

**Q. What happens if you were to use two '>' symbols (i.e. >> mm-first.fa)?**

If you use a single > to a file that already exists it will overwrite the file. If you use two > (i.e. >>) this adds the output of the first section to the end of the file's contents i.e. it appends the content before the >> rather than overwriting the file with said content. So for `mm-first.fa` you will end up with the output above repeated twice.

**Q. How would you determine how many sequences are in the mm-second.fa file?**

The exact same method as before can be used i.e. `grep -c ">" mm-second.fa`. This tells us there are 89 sequences in this file.

## Making plots of summary statistics of the blast search

A subset of the proteins in the mouse genome was blasted against the Zebrafish proteome using the remote server and the results were saved into a blastn format 6 in a .tsv file. This file can then be read into R studio, either on the online (remote) R-studio or a local R-studio.

```r
# Set up
# Load packages
library("ggplot2")

# Read file
blast <- read.table("mm-second.x.zebrafish.tsv", header = FALSE)

# Set column names
colnames(blast) <- c("qseqid", "sseqid", "pident", "length", "mismatch",
"gapopen", "qstart", "qend", "sstart", "send", "evalue", "bitscore")
```

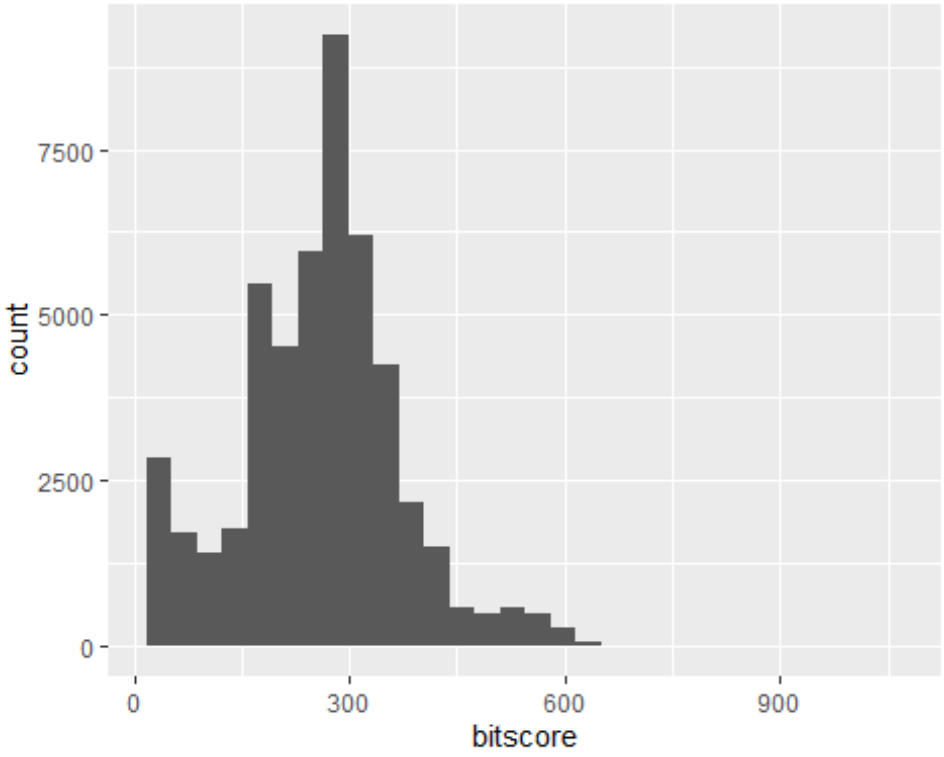
The definition of `bitscore` is as follows:

```
"the size of a sequence database in which the current match could be found
just by chance. The bit-score is a log2 scaled and normalized raw-score. Each
increase by one doubles the required database size (2^bit-score)."
```

We can plot this as a histogram for all the hits. The larger the size of `bitscore` the better, as it suggests that for the current match to happen by chance it has to be a database of a larger size. Note that while in this case bigger is better, in other aspects the `bitscore` is similar to the E value in that both are a metric for the probability that a match is by chance given characterisitcs of the database.

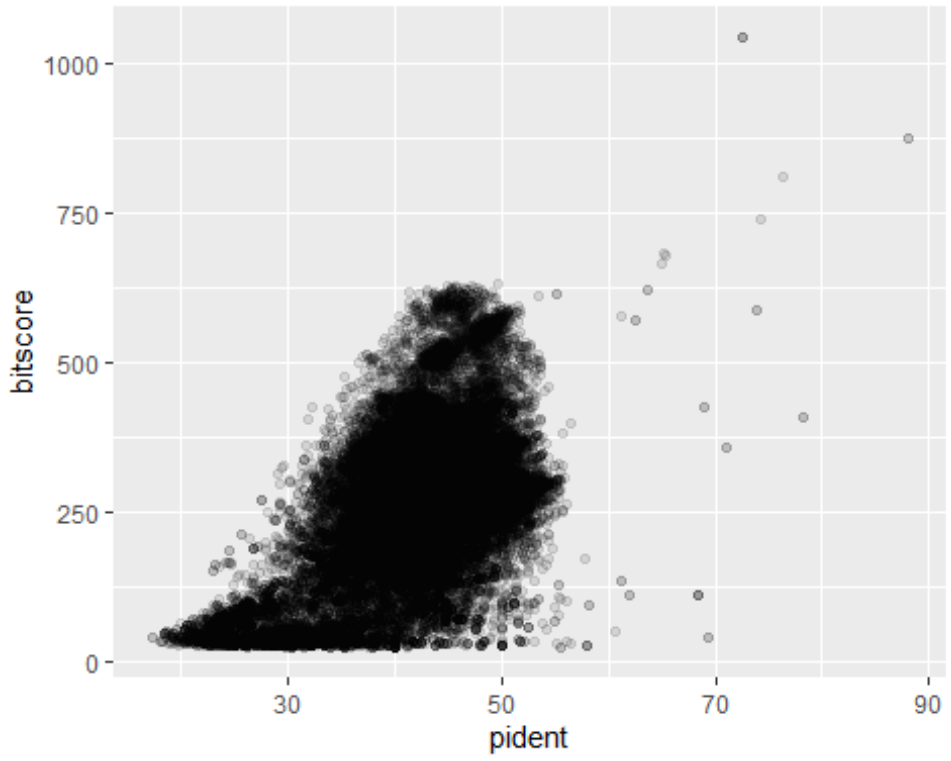
```
# Plot a histogram
ggplot(blast, aes(bitscore)) +
  geom_histogram()

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




It is also possible to plot `bitscore` against `pident`, which is the percentage of identical matches.

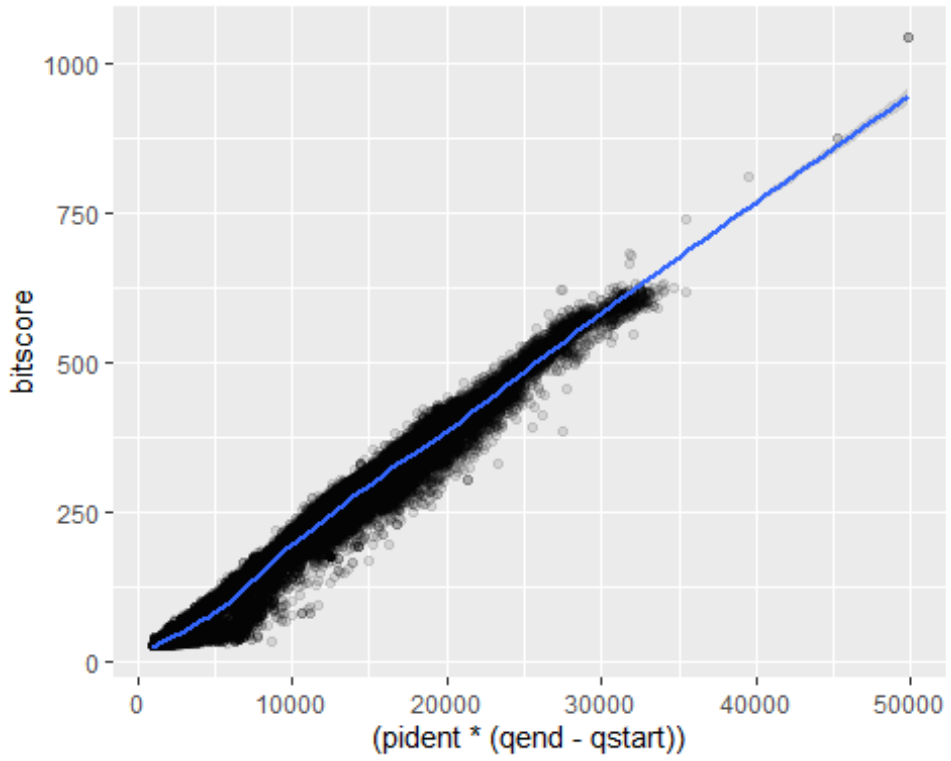
```
# Plot bitscore against pident
ggplot(blast, aes(pident, bitscore)) + geom_point(alpha=0.1)
```




A more accurate summary would be gained from considering not only the percentage identity but the length of the alignment this score is calculated from, thus we can improve the previous plot by plotting percentage identity * length against the Bit score.

```
# Plot percentage identity (pident) * length of alignment against bitscore.
ggplot(blast, aes((pident * (qend - qstart)), bitscore)) +
geom_point(alpha=0.1) + geom_smooth()

## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




## Session Information

```
sessionInfo()

## R version 4.1.2 (2021-11-01)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22000)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United Kingdom.1252
## [2] LC_CTYPE=English_United Kingdom.1252
## [3] LC_MONETARY=English_United Kingdom.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United Kingdom.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] ggplot2_3.3.5
##
## loaded via a namespace (and not attached):
##  [1] highr_0.9         pillar_1.7.0      compiler_4.1.2    tools_4.1.2
##  [5] digest_0.6.29     lattice_0.20-45   nlme_3.1-153      evaluate_0.15
```

```
##  [9] lifecycle_1.0.1  tibble_3.1.6     gtable_0.3.0     mgcv_1.8-38
## [13] pkgconfig_2.0.3  rlang_1.0.1      Matrix_1.3-4     cli_3.2.0
## [17] rstudioapi_0.13  yaml_2.2.2       xfun_0.29        fastmap_1.1.0
## [21] withr_2.4.3      stringr_1.4.0    dplyr_1.0.8      knitr_1.37
## [25] generics_0.1.2   vctrs_0.3.8      grid_4.1.2       tidyselect_1.1.2
## [29] glue_1.6.1       R6_2.5.1         fansi_1.0.2      rmarkdown_2.11
## [33] farver_2.1.0     purrr_0.3.4      magrittr_2.0.2   splines_4.1.2
## [37] scales_1.1.1     ellipsis_0.3.2   htmltools_0.5.2  colorspace_2.0-2
## [41] labeling_0.4.2   utf8_1.2.2       stringi_1.7.6    munsell_0.5.0
## [45] crayon_1.5.0
```