

# SeqFu

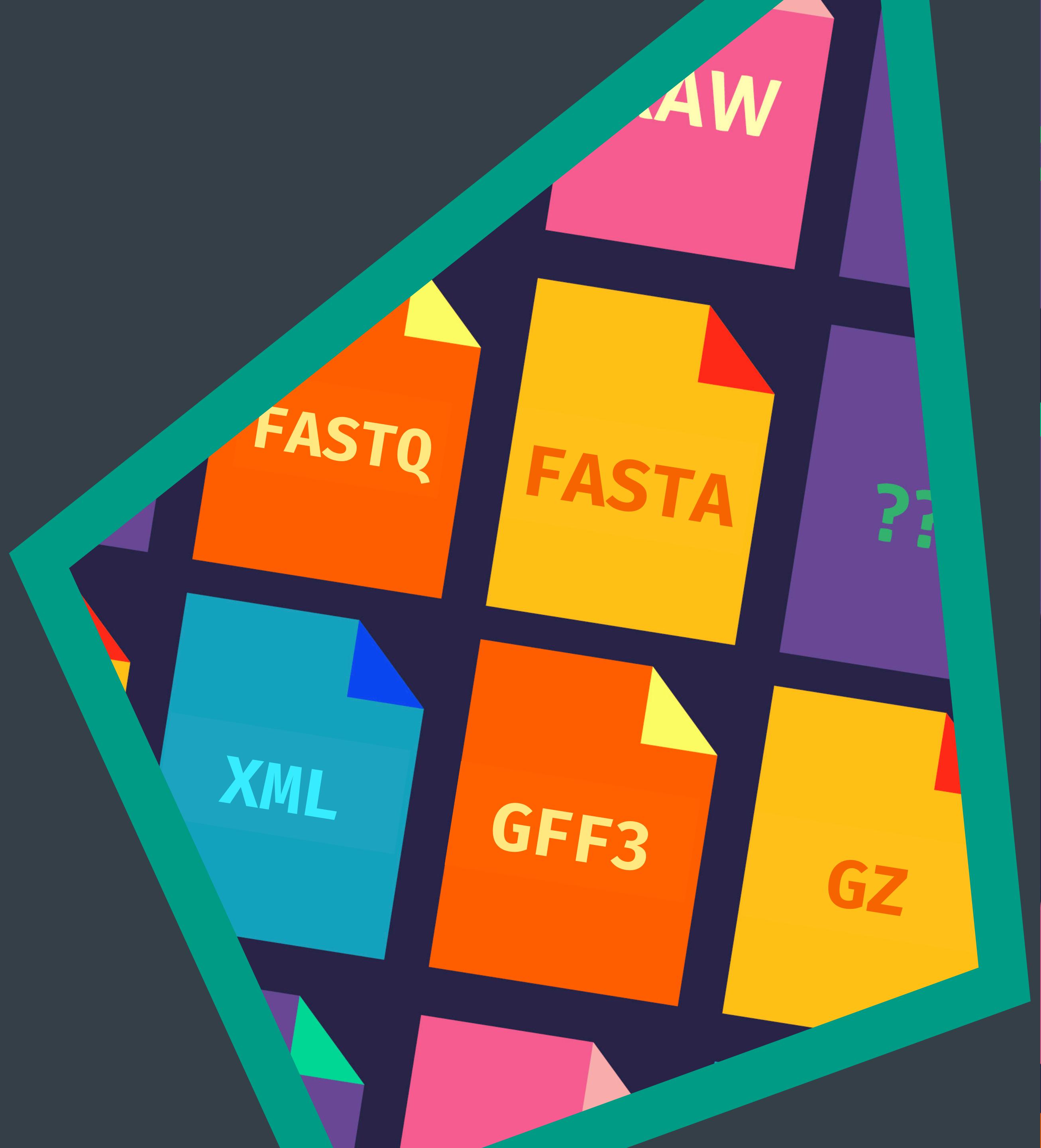
a black belt toolkit  
for sequence  
manipulation



Andrea Telatin



# FASTA.gz and FASTQ.gz



# FASTA

- FASTA to store one or more sequences (DNA, RNA or Protein)
- Determined (assumed correct)

**>SeqOne**

CAGATACGTACGTAGCTTATCTAGA  
CGACTGACTGTACGTACGTACGTACGT  
ACGTACGTAGCTGATCTACGTACGTACGT  
ACGT

**>SeqFu**

CAGATACGTACGTAGCTTATCTAGCAC  
CGACTGACTNA

# FASTQ

- FASTQ to store raw sequencing output
- Associate a quality score to each base

**@Read1**

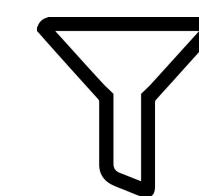
CAGATACGTACGTAGCTTATCTAGCTACGATCTACTAGC  
+  
FEFACFFFFFF,FFF:FFFFFFFFFFFFFF,FFF

**@Read2**

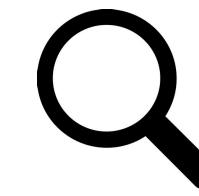
GCTTATCTAGCTACGATCTACTAGCTAACACTCGACAGCNT  
+  
DGGGDGDGDEGGC\*;@BGG?EFFECCG<BGFFFFDBFF!8

# Manipulation from the command line

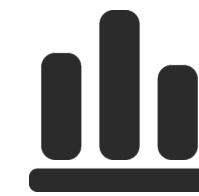
- There are plenty of tasks we need to perform daily on FASTA or FASTQ files:



**Inspection** - peeking into files, checking integrity, oligo/primers...



**Subsetting** - getting some reads from a large dataset



**Summary** - calculating statistics on the assembly/run

You name it: dereplication, reverse complement...

# How it started...



# Daily issues: an example

A run has been **demultiplexed** with the **wrong** sample sheet:

- Each sample received the wrong filename
- It was needed to extract the index from the file content

And many more atomic tasks...

SeqKit didn't always have the answer



classflow — ubuntu@dashboard: ~ — ssh ▾ climb — 86x21

```
(f1) ubuntu@dashboard:~$ seqfu
SeqFu - Sequence Fastx Utilities
version: 1.0.0
```

- count [cnt] : count FASTA/FASTQ reads, pair-end aware
- deinterleave [dei] : deinterleave FASTQ
- derep [der] : feature-rich dereplication of FASTA/FASTQ files
- interleave [ilv] : interleave FASTQ pair ends
- lanes [mrl] : merge Illumina lanes
- sort [srt] : sort sequences by size (uniques)
- stats [st] : statistics on sequence lengths
  
- cat : concatenate FASTA/FASTQ files
- grep : select sequences with patterns
- head : print first sequences
- rc : reverse complement strings or files
- tail : view last sequences
- view : view sequences with color

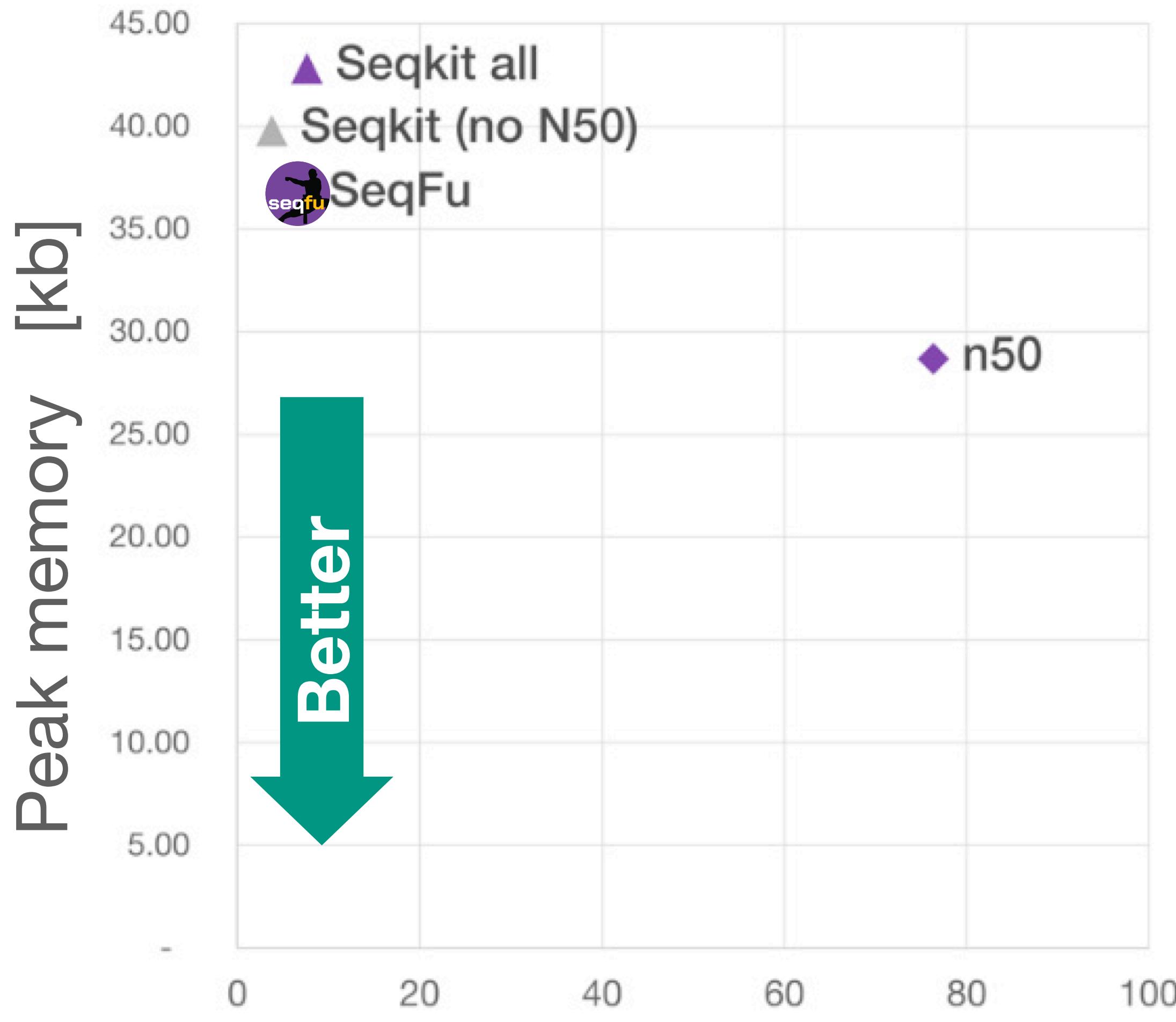
Add --help after each command to print usage

```
(f1) ubuntu@dashboard:~$
```

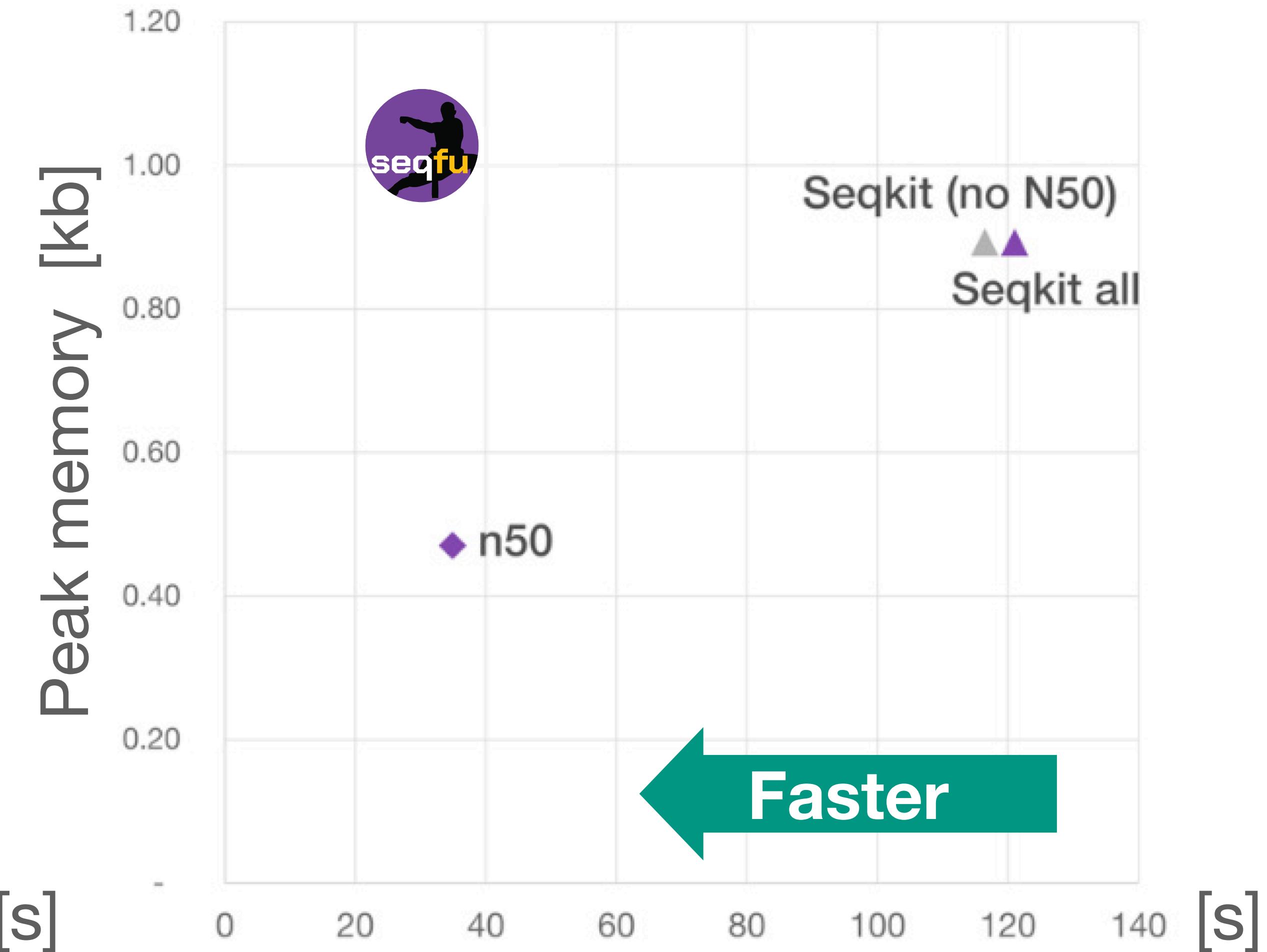


Version 1.0 — 2021

## Reference bacterial genomes



## Human Genome (GRCh38)



**Calculation of N50 and other statistics using two datasets**

# How it is going...





```
$ seqfu  
SeqFu 1.15.3 - FASTX Tools
```



- bases : count bases in FASTA/FASTQ files
- check : check FASTQ file for errors
- count [cnt] : count FASTA/FASTQ reads, pair-end aware
- deinterleave [dei] : deinterleave FASTQ
- derep [der] : feature-rich dereplication of FASTA/FASTQ files
- interleave [ilv] : interleave FASTQ pair ends
- lanes [mrl] : merge Illumina lanes
- list [lst] : print sequences from a list of names
- metadata [met] : print a table of FASTQ reads (mapping files)
- rotate [rot] : rotate a sequence with a new start position
- sort [srt] : sort sequences by size (uniques)
- stats [st] : statistics on sequence lengths
  
- cat : concatenate FASTA/FASTQ files
- grep : select sequences with patterns
- head : print first sequences
- rc : reverse complement strings or files
- tab : tabulate reads to TSV (and viceversa)
- tail : view last sequences
- view : view sequences with colored quality and oligo matches

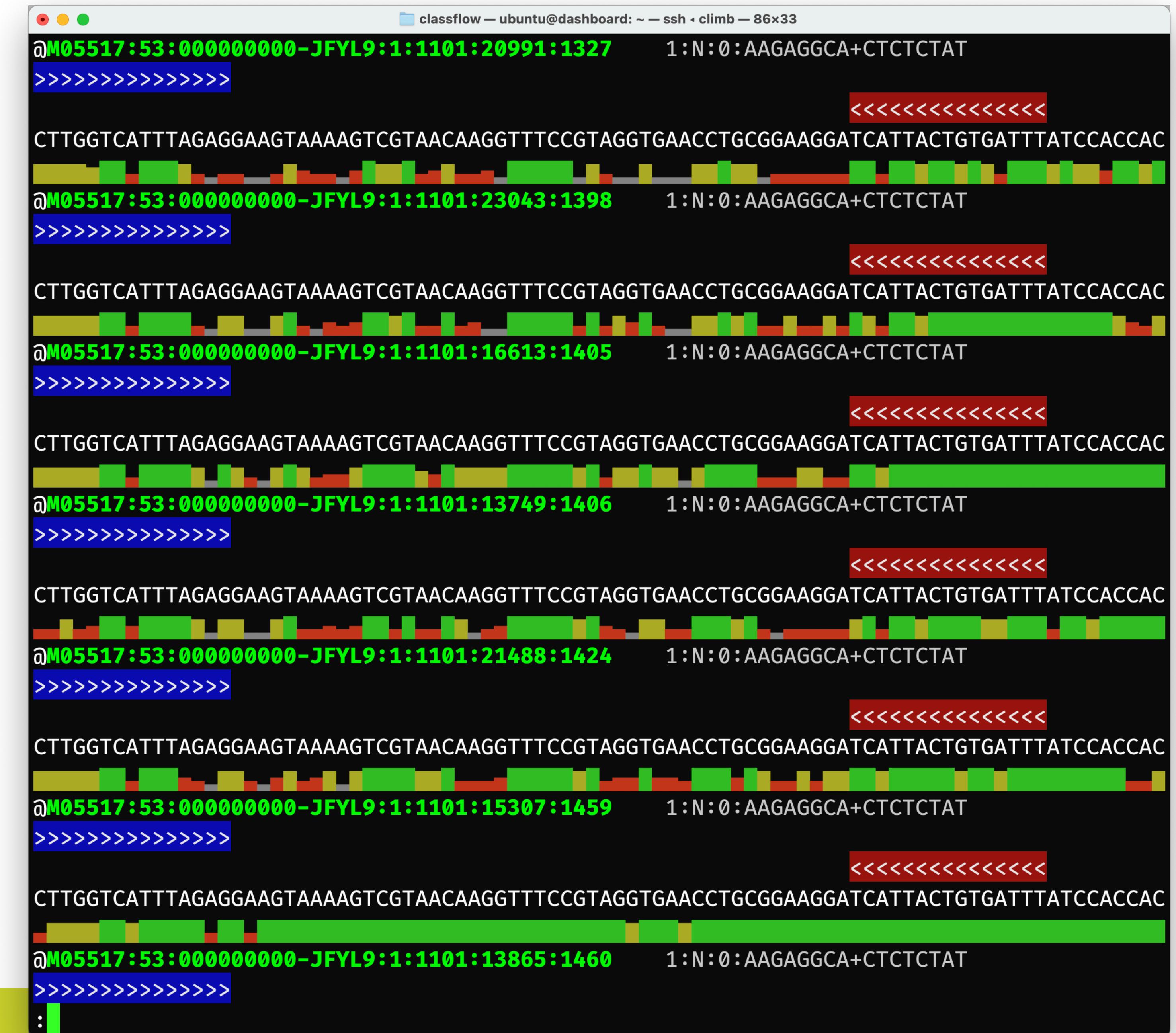
Type 'seqfu version' or 'seqfu cite' to print the version and paper, respectively.  
Add --help after each command to print its usage.

- **19 core functions** in SeqFu 1.17
- **14 utilities** shipped as independent binaries
- **⭐ >70 stars** on GitHub, and good user feedback

Version 1.17 – Dec 2022

# seqfu view

- quickly inspect a FASTQ file, with a coloured representation of qualities
- Degenerated oligo match in both strands supported



# seqfu stats

- Get statistics including N50 from FASTA/FASTQ files
- Nice terminal tables are supported
- Can export in MultiQC format

```
(base) ubuntu@dashboard:~/volume/git/seqfu2/data$ seqfu stats -nt primers/*merge* -b
```

File	#Seq	Total bp	Avg	N50	N75	N90	auN	Min	Max
16S_merge	6,137	2,596,981	423.17	465	449	301	3.08	301	562
16S_vsearch_merge	3,935	1,818,111	462.04	465	465	460	4.02	304	562
its-merge	7,299	1,504,898	206.18	196	196	196	1.05	42	341

```
(base) ubuntu@dashboard:~/volume/git/seqfu2/data$
```

# Good practices

- Documentation
- Streams support
- Automatic tests
- Distribution



The screenshot shows a web browser window with the URL `seqfu.it` in the address bar. The page title is "SeqFu docs" and the sub-page title is "seqfu view". The main content area contains a brief description of the `view` command, its usage, and options. A large code block shows the help output for the `seqfu view` command, detailing options like `-o` for oligo matching, `-r` for second oligo, and `-q` for quality thresholds. Below this, there's a section titled "Example output" with a terminal screenshot showing the command being run on a FASTQ file and the resulting colored bars and arrows.

SeqFu docs

SeqFu 1.16.0 — crayon

Search docs...

1. Installation

2. Overview

3. Short guide

4. About SeqFu

**CORE TOOLS**

seqfu bases

seqfu cat

seqfu check

seqfu count

seqfu deinterleave

seqfu derep

seqfu grep

seqfu head

seqfu interleave

seqfu lanes

seqfu list

seqfu merge

seqfu metadata

seqfu qual

seqfu rc

seqfu rotate

seqfu sort

seqfu stats

## seqfu view

view is one of the core subprograms of SeqFu.

It can be used to visually inspect a FASTQ file printing colored bars for quality scores and highlighting oligonucleotide matches.

```
Usage: view [options] <inputfile> [<input_reverse>]
```

View a FASTA/FASTQ file for manual inspection, allowing to search for an oligonucleotide.

Options:

<code>-o, --oligo1 OLIGO</code>	Match oligo, with ambiguous IUPAC chars allowed (rev. compl. search is performed), color blue
<code>-r, --oligo2 OLIGO</code>	Second oligo to be scanned for, color red
<code>-q, --qual-scale STR</code>	Quality thresholds, seven values separated by columns [default: 3:15:25:28:30:35:40]
<code>--match-ths FLOAT</code>	Oligo matching threshold [default: 0.75]
<code>--min-matches INT</code>	Oligo minimum matches [default: 5]
<code>--max-mismatches INT</code>	Oligo maximum mismatches [default: 2]
<code>--ascii</code>	Use simple ASCII chars instead of UNICODE to render the quality values
<code>-Q, --qual-chars</code>	Show quality characters instead of bars
<code>-n, --nocolor</code>	Disable colored output
<code>--verbose</code>	Show extra information
<code>-h, --help</code>	Show this help

### Example output

The quality scores are rendered as colored bars (grey, red, yellow, green) of different lengths. Matching oligos are rendered as blue arrows (forward) or red arrows (reverse).

```
$ seqfu view data/primers/small.fq -o CCTACGGGAG -r ATGAGCGTCTCG
@M05517:39:000000000-CNNWR:1:1105:16226:1553 1:N:0:GCTCATGA+CGTCTAAAT
>>>>>>>
CCTACGGGAGGCTGCAGAACGCAAGTGGCACAGACGTAAGTTACAGAGCGTCCTTCAGGAAGGGTGCACTTAAAACATATCAATCTGTCT
@M05517:39:000000000-CNNWR:1:1105:22836:1989 1:N:0:GCTCATGA+CGTCTAAAT
>>>>>>>
<<<<<<<<<
CCTACGGGTGGCTGCAGACGAGGATTAGATACCTAGTAGTCCTGTCTCTTACACATCTCCGAGCCCACGAGACGCTCATGAATCTCG
@M05517:39:000000000-CNNWR:1:1105:7693:2054 1:N:0:GCTCACGA+CGTCTAAAT
```

# Documentation

- Extensive online documentation
- Clear help screens from the commands

[telatin.github.io/seqfu2](https://telatin.github.io/seqfu2)

```
data — telatina@N121515: ~/git/seqfu2/data — bash — 91x12
telatina@VM:~/git/seqfu2/data$ seqfu cat --min-len 100 --trim-front 10 dataset.fastq.gz | \
seqfu head -n 100 | \
seqfu stats -n
[seqfu head] Waiting for STDIN... [Ctrl-C to quit, type with --help for info].
[seqfu stats] Waiting for STDIN... [Ctrl-C to quit, type with --help for info].
```

File	#Seq	Total bp	Avg	N50	N75	N90	auN	Min	Max
-	100	24100	241.00	241	241	241	2.41	241	241

telatina@VM:~/git/seqfu2/data\$



# Pipes

- Input/Output streams support to combine multiple tools together

# Automated tests

- Unit tests (164)
- **Continuous integration:** all tests run on the cloud at every change in the code



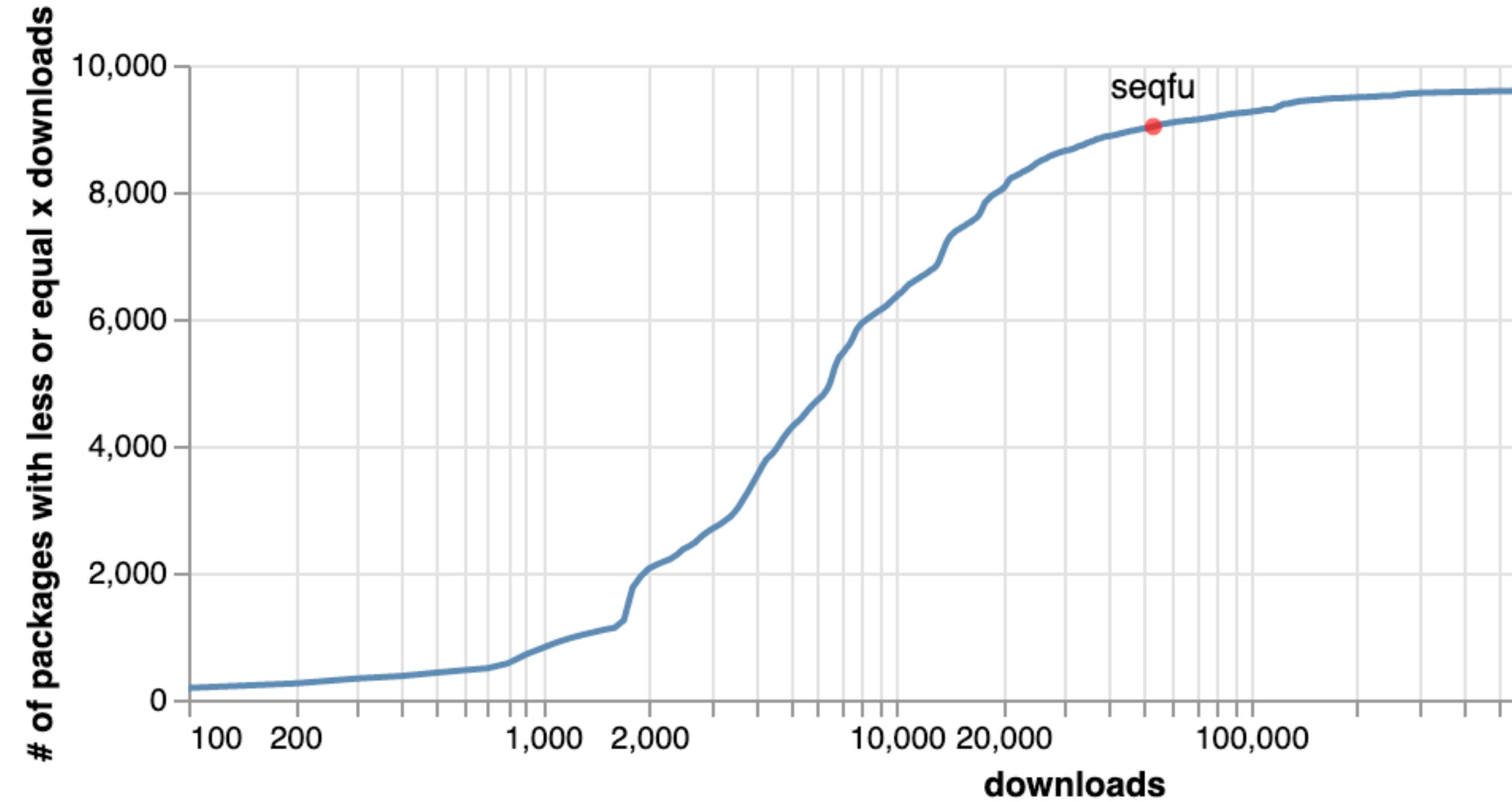
```
classflow — ubuntu@dashboard: ~/volume/git/seqfu2 — ssh ▾ climb — 104x26

Testing module: stats
OK: Checking normal output expecting 2 lines: <2>
OK: Checking normal output expecting total seqs 78730: <78730>
OK: Checking normal output expecting total bases 24299931: <24299931>
OK: Checking normal N50 to be 316: <316>
OK: Checking CSV output N50 is 316, got: <316>
OK: Checking nice output expecting 5 lines: <5>
OK: Checking MultiQC output expecting 39 lines: <39>
OK: Experimental JSON output on 1 line: <1>
OK: Checking default starting by 'filt': <filt>
OK: Checking default N50 starting by 'sort': <sort>
OK: Check absolute paths
OK: Check sort ordered when not specified
OK: Check sort: tot seq sorted at 3300
OK: Check reverse sort: tot seq sorted at 3300
OK: %GC check at 1.00: <1.00>
OK: %GC check at 0.00: <0.00>
OK: %GC check at 0.50: <0.50>
# Finished with 17 passed, 0 failed

Testing module: tab
OK: FASTQ sequence tabulated / detabulated
OK: FASTA sequence tabulated / detabulated
OK: FQ interleaved sequence tabulated / detabulated
```

# BioConda

- Available for **easy** installation from Bioconda
- One of the top downloaded packages in the BioConda repository



```
conda install -c conda-forge -c bioconda seqfu
```

# Acknowledgments

Andrew Page team (Thanh, Sumeet,...)

QIB users for feedback  
(Evelien, Stefano, Rebecca,...)

# Thank you!

for your attention



# Links

- SeqFu Paper  
[mdpi.com/2306-5354/8/5/59](https://mdpi.com/2306-5354/8/5/59)
  - SeqFu repository  
[github.com/telatin/seqfu2](https://github.com/telatin/seqfu2)
  - Info on Core Bioinformatics  
[qintranet.com/bioinformatics/](https://qintranet.com/bioinformatics/)

A] Seq:0:315 Color:Bases  
| 315 : :  
  
.MV .. MV .. F .. CW. PYA .. A.....  
RMVIIMVIAFLICWL PYAGVAFYIFTHC  
RMVIIMVIAFLICWV PYASVAFYIFTHC  
KIMLLVILLFVLSWAPYS AVALVAFAGY  
KISMVIITQFMLSWSPY AII ALLAQFGP  
KIAMTVTCLFIISWSPY AII ALIAQFGP-  
KVALTTISLWFFAWTPYTIINYAGIFES-  
KVALTTISLWFM AWTPYLVICYFGLFKI-  
KIAFVIIIIVFVLSWSPY ACVTTLIAWAGH--  
KVAAMTIGTFMLSWTPY AVVGVFGMIKPHS  
KVALVVILLFIISWSPYSVVALTATAGY--  
KISIVIVTQFLLSWSPY AVVALLAQFGP--  
CMVVIMVICYLLCWLPYGVVALLATFGP--  
RMVIVVMVMAFLLCWL PYSTFALVVATHK--  
IMVLFMVLAFLICWL PYTVFALIVVINP--  
RMVILMVLGFLIAWTPYATVAAWIFFNK--  
RMVVVMVIAFLVCWV PYASVAWXT  
RMVILMVIGFLTAWVPM  
RMVIVVMVVG