Leibniz Institute on Aging –
Fritz Lipmann Institute

Bashbone: NGS Bash library

Konstantin Riege
Computational Biology - Hoffmann Lab

motivation.sh

- you use bash extensively
- you don’t want to re-implement your code in a DSL like CWL/nf/smk

motivation.sh

- wouldn’t it be nice to have

  - an option for in-place parallelization
  - error tracing
  - stdout/stderr logging
  - optional object oriented like syntax
  - proper subprocess termination

```
bash -c 'sleep 10 & sleep 5'
```

motivation.sh

- wouldn’t it be nice to have
  - NGS related functions that
    - use suitable tools and sets of parameters
    - solve tasks efficiently (reduce disk io, max out CPU)
    - cleanup temporary files on success or failure

```
trimmomatic SE -threads $threads $in.gz $out.gz SLIDINGWINDOW:5:20
```

motivation.sh

- wouldn’t it be nice to have
  - NGS related functions that
    - use suitable tools and sets of parameters
    - solve tasks efficiently (reduce disk io, max out CPU)
    - cleanup temporary files on success or failure

```
trimmomatic SE -threads $threads $in.gz $out.gz SLIDINGWINDOW:5:20
```

bypass zlib/gzlib by process substitution

```
trimmomatic SE -threads $threads $in.gz >(pigz -p $threads -c > $out.gz) SLIDINGWINDOW:5:20
```

motivation.sh

- 134 x alignment data wrangling

commander.sh

- bashbone core functions
  - execute commands in parallel instances (locally or HPC)
  - logging and/or benchmark runtime and memory usage
  - inspect jobs and alter task concurrency according to resources availability
  - helper functions
  - error tracing

commander.sh

```
3  for f in *.txt; do
4      grep foo $f | awk '{print $1}'
5  done
```

commander.sh

```
3  for f in *.txt; do
4      grep foo $f | awk '{print $1}'
5  done
```

escape & store cmd

```
3  for f in *.txt; do
4      cmds+=("grep foo $f | awk '{print \$1}'")
5  done
```

commander.sh

```
3  for f in *.txt; do
4      grep foo $f | awk '{print $1}'
5  done
```

escape & store cmd

```
3  for f in *.txt; do
4      cmds+=("grep foo $f | awk '{print \$1}'")
5  done
```

parallelize

```
3  for f in *.txt; do
4      cmds+=("grep foo $f | awk '{print \$1}'")
5  done
6  printf "%s\n" "${cmds[@]}" | parallel --termseq INT,1000,TERM,0 --halt now,fail=1 --line-buffer -P $n -I {} bash -c {}
```

commander.sh

- commander::makecmd
- commander::runcmd

```
3  for f in *.txt; do
4      grep foo $f | awk '{print $1}'
5  done
```

store & parallelize

```
3  for f in *.txt; do
4      commander::makecmd -v f -a cmds -c <<-'CMD'
5          grep foo $f | awk '{print $1}'
6      CMD
7  done
8  commander::runcmd -a cmds -i $n
```

commander.sh

- commander::makecmd
- commander::runcmd

```
3  for f in *.txt; do
4      grep foo $f | awk '{print $1}'
5  done
```

store & parallelize

```
3  for f in *.txt; do
4      commander::makecmd -v f -a cmds -c <<-'CMD'
5          grep foo $f | awk '{print $1}'
6      CMD
7  done
8  commander::runcmd -a cmds -i $n
```

- progress::log monitors a bash function

```
8  progress::log -v 1 -o "$log" -f commander::runcmd -a cmds -i $n
```

commander.sh

- to see available options, use bashbone interactively

turn on benchmarking

name jobs & log single cmd

```
commander::runcmd usage:
-v             | verbose on
-b             | benchmark on
-c <env>       | run with conda
-i <instances> | number of parallel
-t <instances> | obsolete synonym for -i
-s <idx[:idx]> | execute only jobs from cmds array starting from given index or range (default: 1)
-n <name>      | optional. prefix of logs and jobs - should be unique
-o <path>      | optional. for scripts, logs and exit codes
-r             | optional. override existing logs
-a <cmds>      | array of
example:
commander::runcmd -v -b -i 2 -a cmd
example2:
commander::runcmd -i 2 -n current -o ~/jobs -r -a cmd
```

commander.sh

- Local: commander::runcmd commander::runstat commander::runalter

```
NAME        STARTED                   JOBID   MEMORY  ELAPSED  USER    STATE  TASKID
sleep_test  Tue-Feb-7-15:29:59-2023   106029  11324   00:05    kriege  r      9
sleep_test  Tue-Feb-7-15:29:59-2023   106013  11328   00:05    kriege  r      5
sleep_test  Tue-Feb-7-15:29:59-2023   106018  11332   00:05    kriege  r      7
sleep_test  Tue-Feb-7-15:29:59-2023   106024  11336   00:05    kriege  r      8
sleep_test  Tue-Feb-7-15:29:59-2023   106014  11380   00:05    kriege  r      6
sleep_test  Tue-Feb-07-15:29:58-2023  105979  *       *        kriege  q      10-100
```

commander.sh

- Local: commander::runcmd commander::runstat commander::runalter

```
NAME        STARTED                    JOBID   MEMORY  ELAPSED  USER    STATE  TASKID
sleep_test  Tue-Feb-7-15:29:59-2023     106029  11324   00:05    kriege  r      9
sleep_test  Tue-Feb-7-15:29:59-2023     106013  11328   00:05    kriege  r      5
sleep_test  Tue-Feb-7-15:29:59-2023     106018  11332   00:05    kriege  r      7
sleep_test  Tue-Feb-7-15:29:59-2023     106024  11336   00:05    kriege  r      8
sleep_test  Tue-Feb-7-15:29:59-2023     106014  11380   00:05    kriege  r      6
sleep_test  Tue-Feb-07-15:29:58-2023    105979  *       *        kriege  q      10-100
```

- HPC: commander::qsubcmd commander::qstat commander::qalter

```
JOBID   PRIOR    NAME                USER    STATE  STARTED              QUEUE                                SLOTS  TASKID
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:04:10  threads@bcl103.scinet.fli-leibniz.de  56     48
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:20:40  threads@bcl101.scinet.fli-leibniz.de  56     49
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:20:55  threads@bcl108.scinet.fli-leibniz.de  56     50
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:22:55  threads@bcl105.scinet.fli-leibniz.de  56     51
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:22:55  threads@bcl106.scinet.fli-leibniz.de  56     52
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:23:25  threads@bcl109.scinet.fli-leibniz.de  56     53
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:25:10  threads@bcl104.scinet.fli-leibniz.de  56     54
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:32:40  threads@bcl111.scinet.fli-leibniz.de  56     55
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:34:55  threads@bcl110.scinet.fli-leibniz.de  56     56
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:36:55  threads@bcl111.scinet.fli-leibniz.de  56     57
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:39:25  threads@bcl102.scinet.fli-leibniz.de  56     58
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:41:25  threads@bcl110.scinet.fli-leibniz.de  56     59
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:44:40  threads@bcl102.scinet.fli-leibniz.de  56     60
114941  0.55500  all_atac_pub_adjust  kriege  r      2023-02-06T14:46:10  threads@bcl103.scinet.fli-leibniz.de  56     61
114941  0.55500  all_atac_pub_adjust  kriege  qw     2023-02-06T12:13:52  *                                    56     62-65:1
```

helper.sh

- helper::addmemberfunctions for object oriented like syntax

```
3  declare -a arr
4  helper::addmemberfunctions -v arr
5  arr.push foo
6  arr.push bar
7  arr.sort
8  arr.length # 2
9  arr.uc
10 arr.print # BAR FOO
```

- helper::sort
  - parallel sorting a file or stdin

- helper::pgzip
  - parallel compressing a file or stdin
  - gz indexing for random access based on
    - line numbers (pigz + gztool)
    - or byte offset (bgzip + tabix)

activate.sh

- download, source and use

```
git clone --recursive https://github.com/Hoffmann-Lab/bashbone
source ./activate.sh
```

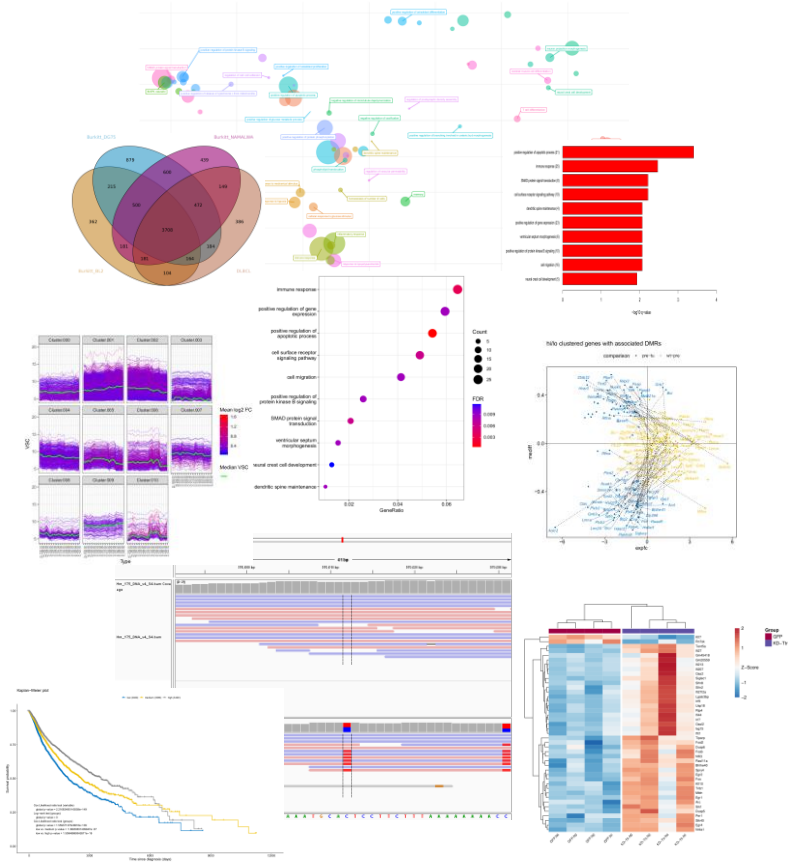
  - interactively with temporary changes to environment and local error tracing
  - in a script with global changes to the environment

```
└─ test::test
:INFO: start testing
error incoming!
:ERROR: in /u/people/kriege/workspace/bash/bashbone/lib/test.sh (function: test::error) @ line 21: echo foo | grep bar
:ERROR: in /u/people/kriege/workspace/bash/bashbone/lib/test.sh (function: test::start) @ line 16: test::error
:ERROR: in /u/people/kriege/workspace/bash/bashbone/lib/test.sh (function: test::test) @ line 10: cat <(test::start)
:ERROR: exit code 143
```

modules.sh

- provision of 83 functions

alignment::add4stats	alignment::addreadgroup	alignment::bamqc
alignment::bqsr	alignment::bulkindex	alignment::bwa
alignment::clipmateoverlaps	alignment::clipmateoverlaps_alt	alignment::inferstrandness
alignment::leftalign	alignment::mkreplicates	alignment::postprocess
alignment::qcstats	alignment::reorder	alignment::rmduplicates
alignment::segemehl	alignment::slice	alignment::soft2hardclip
alignment::splitncigar	alignment::star	alignment::strandsplit
alignment::tobed	bisulfite::bwa	bisulfite::haarz
bisulfite::join	bisulfite::mecall	bisulfite::methyldackel
bisulfite::metilene	bisulfite::mspicut	bisulfite::segemehl
cluster::coexpression	cluster::coexpression_deseq	enrichment::go
expression::deseq	expression::diego	expression::join
expression::join_deseq	fusions::arriba	fusions::join2arriba
fusions::starfusion	genome::indexgtf	genome::mkdict
genome::mkgodb	genome::view	peaks::gem
peaks::gem_idr	peaks::genrich	peaks::genrich_idr
peaks::m6aviewer	peaks::m6aviewer_idr	peaks::macs
peaks::macs_idr	peaks::matk	peaks::matk_idr
peaks::peakachu	peaks::peakachu_idr	preprocess::add4stats
preprocess::cutadapt	preprocess::dedup	preprocess::fastqc
preprocess::qcstats	preprocess::rcorrector	preprocess::rmpolynt
preprocess::sortmerna	preprocess::sortmerna_new	preprocess::trimmomatic
quantify::featurecounts	quantify::tpm	survival::gettcga
survival::ssgsea	variants::bcftools	variants::freebayes
variants::haplotypecaller	variants::makepondb	variants::mutect
variants::panelofnormals	variants::platypus	variants::tree
variants::vardict	variants::vardict_threads	variants::varscan
variants::vcfnorm	visualize::venn	



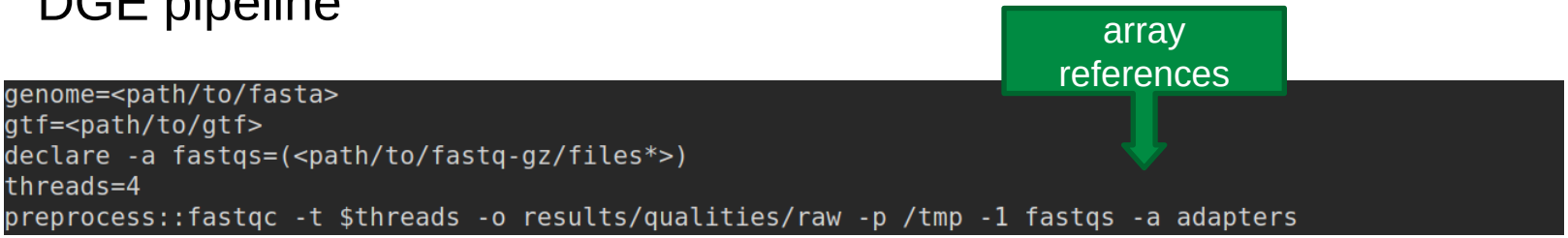
modules.sh

- DGE pipeline


```
genome=<path/to/fasta>
gtf=<path/to/gtf>
declare -a fastqs=(<path/to/fastq-gz/files*>)
threads=4
```

modules.sh

- DGE pipeline



modules.sh

- DGE pipeline

```
genome=<path/to/fasta>
gtf=<path/to/gtf>
declare -a fastqs=(<path/to/fastq-gz/files*>)
threads=4
preprocess::fastqc -t $threads -o results/qualities/raw -p /tmp -1 fastqs -a adapters
preprocess::trimmomatic -t $threads -1 fastqs
preprocess::cutadapt -t $threads -o results/adapterclipped -a adapters -1 fastqs
```

modules.sh

- DGE pipeline

```
genome=<path/to/fasta>
gtf=<path/to/gtf>
decla                /to/fastq-gz/files*>)
threa
preprocess::t                threads -o results/qualities/raw -p /tmp -1 fastqs -a adapters
preprocess::t  mmomatic -t $threads -1 fastqs
preprocess::c  adapt -t $threads -o results/adapterclipped -a adapters -1 fastqs
alignment::segemehl -t $threads -g $genome -o results/mapped -1 fastqs -r mapped
```

segemehl,
star or bwa

modules.sh

- DGE pipeline

```
genome=<path/to/fasta>
gtf=<path/to/gtf>
declare -a fastqs=(<path/to/fastq-gz/files*>)
threads=4
preprocess::fastqc -t $threads -o results/qualities/raw -p /tmp -1 fastqs -a adapters
preprocess::trimmomatic -t $threads -1 fastqs
preprocess::cutadapt -t $threads -o results/adapterclipped -a adapters -1 fastqs
alignment::segemehl -t $threads -g $genome -o results/mapped -1 fastqs -r mapped
alignment::postprocess -j uniqify -t $threads -p /tmp -o results/mapped -r mapped
alignment::postprocess -r sort -t $threads -p /tmp -o results/mapped -r mapped
alignment::inferstrandness -t $threads -g $gtf -p /tmp -r mapped -x strandness
quantify::featurecounts -t $threads -p /tmp -g $gtf -o results/counted -r mapped -x strandness
quantify::tpm -t $threads -g $gtf -o results/counted -r mapped
expression::deseq -t $threads -g $gtf -c comparisons -i results/counted -o results/diffgenes -r mapped
cluster::coexpression -t $threads -g $gtf -i results/counted -o results/coexpressed -r mapped
```

pipelines.sh

- rippchen.sh
  - (co-)expression, splice junction and enrichment analyses from RNAseq
  - gene fusion detection
  - peak calling from ChIPseq, ATACseq, RIPseq
  - methylation analyses from RRBS or WGBS

- muvac.sh
  - somatic or germline variant calling from RNAseq, WES, WGS

```
commander::makecmd -a cmds1 -c <<-CMD
    rippchen.sh
    -v 2 -r -t $threads -mem 10000 -xmem 240000
    -o RRBS -l RRBS/logs/run.log -tmp /data/tmp
    -g $genome -gtf $gtf
    -1 RRBS/R1.list -2 RRBS/R2.list
    -b 0 -no-trim -no-mspi
    -no-bwa -skip md5
CMD
```

setup.sh

- setup conda environments (latest or from yaml files)

  fastqc cutadapt rcorrector star bwa rseqc subread htseq picard bamutil fgbio macs2 genrich peakachu diego gatk4 freebayes varscan igv intervene raxml metilene umitools methyldackel idr sortmerna bwameth vardict snpeff platypus trimmomatic segemehl starfustion gem m6aviewer newicktopdf ssgsea gztool mdless pugz datamash samtools bedtools ucsc-facount khmer htseq bcftools vcflib vt vcftools sra-tools entrez-direct

- download and compile non-conda tools
- setup databases and java wrapper

summary.sh

- you use bash scripts extensively
- you don’t want to re-implement your code to create a workflow
- bashbone offers
  - an option for in-place parallelization
  - error tracing
  - logging
  - subprocess termination
  - object oriented like syntax
  - NGS functions and pipelines

```
for f in *.txt; do
    commander::makecmd -v f -a cmds -c <<-'CMD'
        grep foo $f | awk '{print $1}'
    CMD
done
commander::runcmd -a cmds -i $n
```

```
test::test
:INFO: start testing
error incoming!
:ERROR: in /u/people/kriege/workspace/bash/bashbone/lib/test.sh (function: test::error) @ line 21: echo foo | grep bar
:ERROR: in /u/people/kriege/workspace/bash/bashbone/lib/test.sh (function: test::start) @ line 16: test::error
:ERROR: in /u/people/kriege/workspace/bash/bashbone/lib/test.sh (function: test::test) @ line 10: cat <(test::start)
:ERROR: exit code 143
```

NAME	STARTED	JOBID	MEMORY	ELAPSED	USER	STATE	TASKID
sleep_test	Tue-Feb-7-15:29:59-2023	106029	11324	00:05	kriege	r	9
sleep_test	Tue-Feb-7-15:29:59-2023	106013	11328	00:05	kriege	r	5
sleep_test	Tue-Feb-7-15:29:59-2023	106018	11332	00:05	kriege	r	7
sleep_test	Tue-Feb-7-15:29:59-2023	106024	11336	00:05	kriege	r	8
sleep_test	Tue-Feb-7-15:29:59-2023	106014	11380	00:05	kriege	r	6
sleep_test	Tue-Feb-07-15:29:58-2023	105979	*	*	kriege	q	10-100

```
bash -c '{ trap ":" INT; exec sleep 10; } & sleep 5'
```

```
declare -a arr
helper::addmemberfunctions -v arr
arr.push foo
arr.push bar
arr.sort
arr.length # 2
arr.uc
arr.print # BAR FOO
```

outlook.sh

- support for single cell NGS data
- slurm support
- skip re-calculation if
  - new command is equal to command of existing file
  - checksums of new command input files didn’t change

integrate shournal by Tycho Kirchner

exit.sh

- Steve
- Tycho and beta testers
  - Elina
  - Robert
  - Arne

- CF Life Science Computing
  - Bernd

- CF Next Generation Sequencing
  - Marco

open PhD and post-doc positions available!

https://github.com/Hoffmann-Lab/bashbone

snakemake vs bashbone

```
rule NAME:
    input: "path/to/inputfile", "path/to/other/inputfile"
    output: "path/to/outputfile", "path/to/another/outputfile"
    threads: 8
    message: "Executing somecommand with {threads} threads on the following files {input}."
    shell: "somecommand --threads {threads} {input} {output}"
```

```
#!/usr/bin/env bash

for i in "${!_fq1_bwa[@]}"; do
    o="$(basename "${_fq1_bwa[$i]}")"
    o="$outdir/${o%.*}"
    bwa mem -a -Y -t $threads "$idxprefix" "${_fq1_bwa[$i]}" ${_fq2_bwa[$i]:+"${_fq2_bwa[$i]}"} \
    | samtools view -@ $threads -b > "$o.bam"
done
```

snakemake vs bashbone

```
rule align:
    input:
        reads=["{sample}_R1.fq",
        reference="ref.fa",
    output:
        "{sample}.sam"
    params:
        opts="-M",
    threads: 4
    log:
        "align/{sample}.log"
    conda:
        "envs/align.yaml"
    script:
        "scripts/align.sh"
```

```
#!/usr/bin/env bash

exec 2> "${snakemake_log[0]}"  # send all stderr from

reads=(${snakemake_input[reads]})

r1="${reads[0]}"
r2="${reads[1]}"

bwa index "${snakemake_input[reference]}"
bwa mem ${snakemake_params[opts]} -t ${snakemake[threa
    "${snakemake_input[reference]}" > "${s
```

```
#!/usr/bin/env python3
import os
import sys

from snakemake.util                    ties

jobscript = sys.arg
job_properties = read_job_properties(jobscript)

# do something useful with the threads
threads = job_properties[threads]

# access property defined in the cluster configurat
job_properties["cluster"]["ti

os.system("qsub -t {threads}                    at(threa
```

Python stuff

def. vars

def. error

parse options

def. cmd

```
__author__ = "Johannes Köster, Julian de Ruiter"
__copyright__ = "Copyright 2016, Johannes Köster and Julian de Ruiter"
__email__ = "koester@jimmy.harvard.edu, julianderuiter@gmail.com"
__license__ = "MIT"


import tempfile
from os import path
from snakemake.shell import shell
from snakemake_wrapper_utils.java import get_java_opts
from snakemake_wrapper_utils.samtools import get_samtools_opts


# Extract arguments.
extra = snakemake.params.get("extra", "")
log = snakemake.log_fmt_shell(stdout=False, stderr=True)
sort = snakemake.params.get("sorting", "none")
sort_order = snakemake.params.get("sort_order", "coordinate")
sort_extra = snakemake.params.get("sort_extra", "")
samtools_opts = get_samtools_opts(snakemake)
java_opts = get_java_opts(snakemake)


index = snakemake.input.idx
if isinstance(index, str):
    index = path.splitext(snakemake.input.idx)[0]
else:
    index = path.splitext(snakemake.input.idx[0])[0]


# Check inputs/arguments.
if not isinstance(snakemake.input.reads, str) and len(snakemake.input.reads) not in {
    1,
    2,
}:
    raise ValueError("input must have 1 (single-end) or 2 (paired-end) elements")


if sort_order not in {"coordinate", "queryname"}:
    raise ValueError("Unexpected value for sort_order ({})".format(sort_order))


# Determine which pipe command to use for converting to bam or sorting.
if sort == "none":
    # Simply convert to bam using samtools view.
    pipe_cmd = "samtools view {samtools_opts}"

elif sort == "samtools":
    # Add name flag if needed.
    if sort_order == "queryname":
        sort_extra += " -n"

    # Sort alignments using samtools sort.
    pipe_cmd = "samtools sort {samtools_opts} {sort_extra} -T {tmpdir}"

elif sort == "picard":
    # Sort alignments using picard SortSam.
    pipe_cmd = "picard SortSam {java_opts} {sort_extra} --INPUT /dev/stdin --TMP_DIR {tmpdir} --SORT_ORD

else:
    raise ValueError(f"Unexpected value for params.sort ({sort})")

with tempfile.TemporaryDirectory() as tmpdir:
    shell(
        "(bwa mem"
        " -t {snakemake.threads}"
        " {extra}"
        " {index}"
        " {snakemake.input.reads}"
        " | " + pipe_cmd + ") {log}"
    )
```

snakemake vs bashbone

def. error

parse options

def. vars

Bash stuff

def. cmd

```
3  set -e
4  function bwa_mem(){
5      local OPTIND arg mandatory threads genome idxprefix outdir accuracy=95 readlength=100
6      declare -n _fq1_bwa _fq2_bwa
7      while getopts 't:g:x:a:l:o:1:2:' arg; do
8          case $arg in
9              t)  ((++mandatory)); threads=$OPTARG;;
10             g)  ((++mandatory)); genome="$OPTARG";;
11             x)  ((++mandatory)); idxprefix="$OPTARG";;
12             o)  ((++mandatory)); outdir="$OPTARG/bwa"; mkdir -p "$outdir";;
13             a)  accuracy=$OPTARG;;
14             l)  length=$OPTARG;;
15             1)  ((++mandatory)); _fq1_bwa=$OPTARG;;
16             2)  _fq2_bwa=$OPTARG;;
17             *)  false;;
18         esac
19     done
20     [[ $# -lt 4 ]]
21
22     local hosts=$(pssh -i -H "hosts.txt" "grep -c processor /proc/cpuinfo" \
23         | paste - - | awk -v t=$threads '{printf "%0d/%s\n",$NF/t,$(NF-1)}' | xargs echo | tr ' ' ',')
24     # 1/bcl101,2/bcl102,2/bcl103,1/bcl104
25
26     for i in "${!_fq1_bwa[@]}"; do
27         o="$(basename "${_fq1_bwa[$i]}")"
28         o="$outdir/${o%.*}"
29         echo "
30             bwa mem -a -Y -t $threads '$idxprefix' '${_fq1_bwa[$i]}' ${_fq2_bwa[$i]:+'${_fq2_bwa[$i]}'}
31             | samtools view -@ $threads -b > '$o.bam'
32         "
33     done | parallel --sshdelay 0.1 -I {} -S "$hosts" bash -c {}
34 }
```

snakemake vs bashbone

	snakemake	bashbone
workload managers	sge, slurm	sge
cloud support	docker	bio.container
pipes	command groups	process substitution
modules	130	80
skip	time stamps from defined input files	shournal query by output file
language	python + yaml + bash	bash