Analysis of unmapped P. major reads

Aims

Create indeces for bam-files

I created indeces for bam-files of all samples with the shell script `create_indeces_for bams.sh`:

```
#!/bin/bash



for file in *.bam
do
    samtools index -@ 4 $file
done
```

Subsetting of bam files

Made subsets of bam files for testing using subset_bams.sh:

```
#!/bin/bash

conda activate samtools

mkdir subsets

for file in *.bam
do
    FILENAME="$file"
    FILENAME=${FILENAME%.bam*}
    echo $FILENAME
    samtools view -bo "$FILENAME"_subset.bam -s 123.001 "$file"
done

conda deactivate

# move subset-bams into the subset directory
mv *_subset.bam subsets/
```

Setup miniconda environments

Look at file sizes and raw data more generally

Count number of reads in bam files

I counted the number of reads in the bam files with the shell-script `num_reads_bams.sh`:

```
#!/bin/bash


touch num_reads.txt

for file in *.bam
do
    printf '%s\t%s\n' $file $(samtools view -@ 8 -c $file)

done > num_reads.txt
```

Distribution of read lengths

Counts of read lengths can be calculated with the following shell command.

```
# get counts (2. column) of read of different length (1. column)
samtools stats S1_EKDN230004350-
1A_HNW2NDSX_sorted_dedup_unmapped_subset.bam | grep ^RL | cut -f 2-
```

I wrote a Python-script to plot a histogram of the read length of sample bam file:

```
# Plot histograms of read length distribution of sample bam files


import subprocess

import matplotlib.pyplot as plt

shell_command = 'ls | grep .bam'

OUTPUT_STREAM = subprocess.run(
    shell_command, capture_output=True, shell=True, text=True, check=True)

bam_files = OUTPUT_STREAM.stdout.split('\n')[:-1]

#
fig, axs = plt.subplots(4, 5, sharey=True, sharex=True)


positions = {0: (0, 0),
             1: (0, 1),
             2: (0, 2),
             3: (0, 3),
```

```
              4: (0, 4),
              5: (1, 0),
              6: (1, 1),
              7: (1, 2),
              8: (1, 3),
              9: (1, 4),
              10: (2, 0),
              11: (2, 1),
              12: (2, 2),
              13: (2, 3),
              14: (2, 4),
              15: (3, 0),
              16: (3, 1),
              17: (3, 2),
              18: (3, 3),
              19: (3, 4)}


for i, file in enumerate(bam_files):

    sample = file.split('_')[0]
    samtools_command = f'samtools stats {file} | grep ^RL | cut -f 2-
'.format(
        file)

    OUTPUT_STREAM = subprocess.run(
        samtools_command, capture_output=True, shell=True, text=True,
check=True)

    rows = OUTPUT_STREAM.stdout
    split_rows = rows.split('\n')
    split_split_rows = [row.split('\t') for row in split_rows][:-1]
    read_lens = ([int(entry[0]) for entry in split_split_rows])
    read_lens.append(151)
    read_counts = ([int(entry[1]) for entry in split_split_rows])
    axs[positions[i]].stairs(read_counts, read_lens, fill=True)
    axs[positions[i]].set_title(sample, pad=3.0)
    plt.yscale('log')

fig.text(0.5, 0.04, 'Read length', ha='center', va='center', fontsize=18)
fig.text(0.06, 0.5, 'Read counts', ha='center',
         va='center', rotation='vertical', fontsize=18)
plt.show()

print('done')
```

Kraken2 analysis

Get fasta-files from bam-files

For downstream analysis I converted bam-files to fasta-files with `convert_bams_to_fastas.sh`:

```
#!/bin/bash



for file in *.bam
do
    FILENAME="$file"
    FILENAME=${FILENAME%.bam*}
    echo $FILENAME

    samtools fasta -@ 4 $file > "$FILENAME".fasta
done
```

Download additional genomes

I downloaded additional genomes of Parus major, Gallus gallus, Haemoproteus tartakovskyi, blue tit, Tibethan ground tit and zebra finch.

```
#!/bin/bash


# Parus major
ncbi-genome-download --section genbank vertebrate_other -A GCA_001522545.3
-F fasta,assembly-report -p 4 -r 3 -o /work/mnikvell/data/genomes/
gzip -d
/work/data/genomes/genbank/vertebrate_other/GCA_001522545.3/GCA_001522545.3
_Parus_major1.1_genomic.fna.gz

# Gallus gallus
ncbi-genome-download --section genbank vertebrate_other -A GCA_016699485.1
-F fasta,assembly-report -p 4 -r 3 -o /work/mnikvell/data/genomes/
gzip -d
/work/data/genomes/genbank/vertebrate_other/GCA_016699485.1/GCA_016699485.1
_bGalGal1.mat.broiler.GRCg7b_genomic.fna.gz

# Haemoproteus tartakovskyi
ncbi-genome-download --section genbank invertebrate -A GCA_001625125.1 -F
fasta,assembly-report -p 4 -r 3 -o /work/mnikvell/data/genomes
gzip -d
/work/mnikvell/data/genomes/genbank/protozoa/GCA_001625125.1/GCA_001625125.
1_ASM162512v1_genomic.fna.gz

# blue tit genome
ncbi-genome-download --section genbank vertebrate_other -A GCA_002901205.1
-F fasta,assembly-report -p 4 -r 3 -P -o /work/mnikvell/data/genomes/

# ground tit genome
ncbi-genome-download --section genbank vertebrate_other -A GCA_000331425.1
```

```
-F fasta,assembly-report -p 4 -r 3 -P -o /work/mnikvell/data/genomes/

# zebra finch genome
ncbi-genome-download --section genbank vertebrate_other -A GCA_003957565.4
-F fasta,assembly-report -p 4 -r 3 -P -o /work/mnikvell/data/genomes/
```

In order to "be more modular" and make sending job to lido-cluster easier, I divided the building of the kraken and braken dbs and the classification of samples into several steps that could be sent off to the cluster as separate jobs.

Shell script to download up-to-date taxonomy and libraries on lido3-cluster

```
#!/bin/bash -l
#SBATCH --partition=long
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=20:00:00
#SBATCH --cpus-per-task=20
#SBATCH --mem-per-cpu=2G
#SBATCH --job-name=kraken_download_libs_job
#SBATCH --mail-user=nikolas.vellnow@tu-dortmund.de
#SBATCH --mail-type=All

conda activate kraken

THREAD_NUM=20
FOLDER_NAME=full_libs_downloaded
FOLDER_PATH=/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${FOLDER_NAME}/
OUT_PATH=/work/mnikvell/data/Kraken2/dbs/

echo "folder name: ${FOLDER_NAME}"
echo "folder path: ${FOLDER_PATH}"
echo "output path: ${OUT_PATH}"

# create directories in scratch-dir
rm -rf /scratch/mnikvell/kraken_job_${SLURM_JOBID}/
mkdir -p /scratch/mnikvell/kraken_job_${SLURM_JOBID}/
mkdir -p /scratch/mnikvell/kraken_job_${SLURM_JOBID}/${FOLDER_NAME}


# scratch directory
echo "content of scratch dir: $(ls -R /scratch/mnikvell/)"

# move to job directory
cd /scratch/mnikvell/kraken_job_${SLURM_JOBID}/


# download taxonomy
kraken2-build --download-taxonomy --threads ${THREAD_NUM} --db
"${FOLDER_PATH}"

# download most dbs
```

```
echo 'archaea'
kraken2-build --download-library archaea --threads ${THREAD_NUM} --db
"${FOLDER_PATH}"

echo 'bacteria'
kraken2-build --download-library bacteria --threads ${THREAD_NUM} --db
"${FOLDER_PATH}"

echo 'plasmid'
kraken2-build --download-library plasmid --threads ${THREAD_NUM} --db
"${FOLDER_PATH}"

echo 'viral'
kraken2-build --download-library viral --threads ${THREAD_NUM} --db
"${FOLDER_PATH}"

echo 'human'
kraken2-build --download-library human --threads ${THREAD_NUM} --db
"${FOLDER_PATH}"

echo 'fungi'
kraken2-build --download-library fungi --threads ${THREAD_NUM} --db
"${FOLDER_PATH}"

echo 'plant'
kraken2-build --download-library plant --threads ${THREAD_NUM} --db
"${FOLDER_PATH}"

echo 'protozoa'
kraken2-build --download-library protozoa --threads ${THREAD_NUM} --db
"${FOLDER_PATH}"

echo 'UniVec_Core'
kraken2-build --download-library UniVec_Core --threads ${THREAD_NUM} --db
"${FOLDER_PATH}"

# check what folders are there now
echo "content of folder with transferred data in scratch dir: $(ls
/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${FOLDER_NAME}/)"

# copy outputs back to
cp -a $FOLDER_PATH $OUT_PATH

rm -rf /scratch/mnikvell/kraken_job_${SLURM_JOBID}/

conda deactivate
```

Shell script to add downloaded genomes to libraries on lido3-cluster

```
#!/bin/bash -l
#SBATCH --partition=med
#SBATCH --nodes=1
```

```
#SBATCH --ntasks-per-node=1
#SBATCH --time=4:00:00
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=2G
#SBATCH --job-name=kraken_add_genomes_job
#SBATCH --mail-user=nikolas.vellnow@tu-dortmund.de
#SBATCH --mail-type=All

conda activate kraken

THREAD_NUM=32
SOURCE_NAME=/home/mnikvell/Desktop/work/data/Kraken2/dbs/full_libs_download
ed/
FOLDER_NAME=full_libs_added_genomes
FOLDER_PATH=/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${FOLDER_NAME}/
OUT_PATH=/work/mnikvell/data/Kraken2/dbs/

echo "folder name: ${FOLDER_NAME}"
echo "folder path: ${FOLDER_PATH}"
echo "output path: ${OUT_PATH}"

# create directories in scratch-dir (and delete previous ones)
rm -rf /scratch/mnikvell/kraken_job_${SLURM_JOBID}/
mkdir -p /scratch/mnikvell/kraken_job_${SLURM_JOBID}/

# copy already downloaded libraries to scratch-dir
cp -r $SOURCE_NAME $FOLDER_PATH

# scratch directory
echo "content of mnikvell in scratch dir: $(ls /scratch/mnikvell/)"
echo "content of folder with transferred data in scratch dir: $(ls
/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${FOLDER_NAME}/)"

# move to job directory
cd /scratch/mnikvell/kraken_job_${SLURM_JOBID}/


# add genomes (already downloaded) to library
echo 'genome chicken'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_016699485.1/GCA_01
6699485.1_bGalGal1.mat.broiler.GRCg7b_genomic.fna \
--db "${FOLDER_PATH}" --threads ${THREAD_NUM}

echo 'genome great tit'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_001522545.3/GCA_00
1522545.3_Parus_major1.1_genomic.fna \
--db "${FOLDER_PATH}" --threads ${THREAD_NUM}

echo 'genome blue tit'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_002901205.1/GCA_00
2901205.1_cyaCae2_genomic.fna \
--db "${FOLDER_PATH}" --threads ${THREAD_NUM}
```

```
echo 'genome zebra finch'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_003957565.4/GCA_00
3957565.4_bTaeGut1.4.pri_genomic.fna \
--db "${FOLDER_PATH}" --threads ${THREAD_NUM}

echo 'genome Tibetan ground-tit'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_000331425.1/GCA_00
0331425.1_PseHum1.0_genomic.fna \
--db "${FOLDER_PATH}" --threads ${THREAD_NUM}

echo 'genome blood parasite'
kraken2-build --add-to-library
/work/mnikvell/data/genomes/genbank/protozoa/GCA_001625125.1/GCA_001625125.
1_ASM162512v1_genomic.fna \
--db "${FOLDER_PATH}" --threads ${THREAD_NUM}


# check what folders are there now
echo "content of folder with transferred data in scratch dir: $(ls
/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${FOLDER_NAME}/)"

# copy outputs back to
cp -a $FOLDER_PATH $OUT_PATH

rm -rf /scratch/mnikvell/kraken_job_${SLURM_JOBID}/

conda deactivate
```

Shell script to build kraken db on lido3-cluster

```
#!/bin/bash -l
#SBATCH --partition=long
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=2-00:00:00
#SBATCH --cpus-per-task=40
#SBATCH --mem-per-cpu=6G
#SBATCH --job-name=kraken_build_job
#SBATCH --mail-user=nikolas.vellnow@tu-dortmund.de
#SBATCH --mail-type=All

conda activate kraken

THREAD_NUM=40
SOURCE_NAME=/home/mnikvell/Desktop/work/data/Kraken2/dbs/full_libs_added_ge
nomes/
DB_NAME=full_5_birds_kraken_new
DB_PATH=/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${DB_NAME}/
OUT_PATH=/work/mnikvell/data/Kraken2/dbs/
```

```
echo "db name: ${DB_NAME}"
echo "db path: ${DB_PATH}"
echo "output path: ${OUT_PATH}"


# create directories in scratch-dir
mkdir -p /scratch/mnikvell/kraken_job_${SLURM_JOBID}/

# copy library with added genomes into scratch-dir
cp -R $SOURCE_NAME $DB_PATH

# scratch directory
echo "content of mnikvell in scratch dir: $(ls /scratch/mnikvell/)"
echo "content of folder with transferred data in scratch dir: $(ls
/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${DB_NAME}/)"

# move to job directory
cd /scratch/mnikvell/kraken_job_${SLURM_JOBID}/

# build database
kraken2-build --build --db "${DB_PATH}" --threads ${THREAD_NUM}

# check what folders are there now
echo "content of folder with build db in scratch dir: $(ls
/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${DB_NAME}/)"


# copy outputs back to
cp -a $DB_PATH $OUT_PATH

rm -rf /scratch/mnikvell/kraken_job_${SLURM_JOBID}/

conda deactivate
```

Shell script to send slurm jobs to lido-cluster

I used a script to send a slurm job for each sample to the cluster which then can be classified by kraken and bracken

```
#!/bin/bash

INPUT_PATH=$1

SCRIPT_PATH=${PWD}

cd ${INPUT_PATH}
for file in *_unmapped.fasta
do
    sbatch "${SCRIPT_PATH}/job_script_kraken.sh" ${file}
```

```
done
```

For each of those jobs the following script was send to classify the sample.

Shell script to classify sample with kraken

```
#!/bin/bash -l
#SBATCH --partition=short
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=01:30:00
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=6G
#SBATCH --job-name=kraken_job
#SBATCH --mail-user=nikolas.vellnow@tu-dortmund.de
#SBATCH --mail-type=All

conda activate kraken

FILE_NAME=$1
DB_NAME=full_5_birds
DB_PATH=/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${DB_NAME}/
OUT_PATH=/scratch/mnikvell/kraken_job_${SLURM_JOBID}/kraken_outputs_${SLURM
_JOBID}/
FILE_PATH=/work/mnikvell/data/unmapped_reads/

echo "file name: ${FILE_NAME}"
echo "db name: ${DB_NAME}"
echo "db path: ${DB_PATH}"
echo "output path: ${OUT_PATH}"
echo "file path: ${FILE_PATH}"

# create directories in scratch-dir
rm -rf /scratch/mnikvell/kraken_job_${SLURM_JOBID}/
mkdir -p /scratch/mnikvell/kraken_job_${SLURM_JOBID}/
mkdir -p
/scratch/mnikvell/kraken_job_${SLURM_JOBID}/kraken_outputs_${SLURM_JOBID}/
mkdir -p /scratch/mnikvell/kraken_job_${SLURM_JOBID}/${DB_NAME}


# move database to scratch-dir
cp -a -v "/work/mnikvell/data/Kraken2/dbs/${DB_NAME}/."
"/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${DB_NAME}/"
echo "content of job dir: $(ls
/scratch/mnikvell/kraken_job_${SLURM_JOBID}/)"


# move to job directory
cd /scratch/mnikvell/kraken_job_${SLURM_JOBID}/
```

```
OUTPUT_NAME=output_${FILE_NAME%.*}_${DB_NAME}
echo "output name: ${OUTPUT_NAME}"
CLASSIFIED_NAME=classified_${FILE_NAME%.*}_${DB_NAME}.fasta
echo "classified output name: ${CLASSIFIED_NAME}"
UNCLASSIFIED_NAME=unclassified_${FILE_NAME%.*}_${DB_NAME}.fasta
echo "unclassified output name: ${UNCLASSIFIED_NAME}"
REPORT_NAME=report_${FILE_NAME%.*}_${DB_NAME}
echo "report name: ${REPORT_NAME}"


# move file to scratch-dir
cp -v ${FILE_PATH}${FILE_NAME} /scratch/mnikvell/kraken_job_${SLURM_JOBID}/


kraken2 \
--db ${DB_PATH} \
--output ${OUT_PATH}${OUTPUT_NAME} \
--use-names \
--report ${OUT_PATH}${REPORT_NAME} \
--classified-out ${OUT_PATH}${CLASSIFIED_NAME} \
--unclassified-out ${OUT_PATH}${UNCLASSIFIED_NAME} \
--confidence 0.1 \
--threads 32 \
${FILE_NAME}

# delete fasta-file from scratch dir after classifying it
rm "/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${FILE_NAME}"

cd ${OUT_PATH}
# zip large files
gzip output*
gzip classified*
gzip unclassified*

# copy outputs back to
cp -a "${OUT_PATH}."
/"work/mnikvell/data/unmapped_reads/kraken_outputs_${DB_NAME}_db/"
rm -rf /scratch/mnikvell/kraken_job_${SLURM_JOBID}/

conda deactivate
```

De novo assembly analysis

Separating paired reads

To make use of the paired-end data in the downstream analysis I separated the reads in bam-files intofq-files with either paired1-reads, paired2-reads or singletons witht he following script `separate_paired_reads_from_bam.sh`:

```
```

```
#!/bin/bash


SCRIPT_PATH=${PWD}

# input path to directory with sample .fq-files
DATA_PATH=$1

cd ${DATA_PATH}



for file in *.bam
do
    FILENAME="$file"
    FILENAME=${FILENAME%.bam*}
    echo ${FILENAME}
    samtools collate -u -@ 4 -O ${file} | samtools fastq -@ 4 -1
"${FILENAME}_paired1.fq.gz" -2 "${FILENAME}_paired2.fq.gz" -s
${FILENAME}_"singletons.fq.gz"
done
```

Finding best parameters for assembly

I sent jobs with a range of values for the parameters *k* and *kc* to lido using the script `send_abyss_k_jobs.sh`:

```
#!/bin/bash


SCRIPT_PATH=${PWD}


for kc in 2 3; do

    for k in `seq 70 5 95`;do
        sbatch "${SCRIPT_PATH}/job_script_abyss_test_k.sh" ${k} ${kc}
    done
done
```

For each parameter combination an assembly of sample S1 was performed with abyss using the script `job_script_abyss_test_k.sh`:

```

```

```
#!/bin/bash -l
#SBATCH --partition=med
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=02:59:00
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=500M
#SBATCH --job-name=abyss_job
#SBATCH --mail-user=nikolas.vellnow@tu-dortmund.de
#SBATCH --mail-type=All

conda activate assembly

# hand over parameters for assembly algorithm
K=$1
KC=$2

#FILE_NAME=$1
SAMPLE_NAME=S1
JOB_PATH=/scratch/mnikvell/abyss_job_${SLURM_JOBID}/
IN_PATH=/work/mnikvell/data/unmapped_reads/
OUT_PATH=/work/mnikvell/data/unmapped_reads/${SAMPLE_NAME}_k${K}_kc${KC}

SINGLETONS=${SAMPLE_NAME}_singletons.fq.gz
PAIRED1=${SAMPLE_NAME}_paired1.fq.gz
PAIRED2=${SAMPLE_NAME}_paired2.fq.gz


# create directories in scratch-dir
rm -rf ${JOB_PATH}
mkdir -p ${JOB_PATH}


# move fasta files to scratch-dir
cp -a -v "${IN_PATH}${SINGLETONS}" ${JOB_PATH}
cp -a -v "${IN_PATH}${PAIRED1}" ${JOB_PATH}
cp -a -v "${IN_PATH}${PAIRED2}" ${JOB_PATH}

echo "content of job dir: $(ls ${JOB_PATH})"


# move to job directory
cd ${JOB_PATH}

abyss-pe \
name=${SAMPLE_NAME}\
j=32 \
k=${K} \
kc=${KC} \
B=6G \
in='${PAIRED1} ${PAIRED2}' \
se=${SINGLETONS}


echo "content of dir with results: $(ls ${JOB_PATH})"
```

```
# delete fasta-files from scratch dir after assembly
rm -rf "${JOB_PATH}${SINGLETONS}"
rm -rf "${JOB_PATH}${PAIRED1}"
rm -rf "${JOB_PATH}${PAIRED2}"


# copy output back to work dir
mkdir -p "${OUT_PATH}"
cp -a "${JOB_PATH}." "${OUT_PATH}"
rm -rf ${JOB_PATH}

conda deactivate
```

Shell script to assemble samples

I used a shell script `send_abyss_assembly_jobs.sh` to send each sample to the lido-cluster for assembly:

```
#!/bin/bash


SCRIPT_PATH=${PWD}

# input path to directory with sample .fq-files
DATA_PATH=$1

k=85
kc=2

cd ${DATA_PATH}

for file in *.bam
do
    sbatch "${SCRIPT_PATH}/job_script_abyss_assembly.sh" ${k} ${kc} ${file}
${DATA_PATH}

done
```

Each sample was assembled on lido with the following script `job_script_abyss_assembly.sh`:

```
#!/bin/bash -l
#SBATCH --partition=med
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=02:59:00
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=500M
#SBATCH --job-name=abyss_job
```

```
#SBATCH --mail-user=nikolas.vellnow@tu-dortmund.de
#SBATCH --mail-type=All

conda activate assembly

# hand over parameters for assembly algorithm
K=$1
KC=$2

# hand over file name (bam-file)
FILE_NAME=$3
SAMPLE_NAME=${FILE_NAME%.bam*}
JOB_PATH=/scratch/mnikvell/abyss_job_${SLURM_JOBID}/
IN_PATH=$4
OUT_PATH=${IN_PATH}${SAMPLE_NAME}_k${K}_kc${KC}

SINGLETONS=${SAMPLE_NAME}_singletons.fq.gz
PAIRED1=${SAMPLE_NAME}_paired1.fq.gz
PAIRED2=${SAMPLE_NAME}_paired2.fq.gz


echo "file name: ${FILE_NAME}"
echo "SAMPLE_NAME: ${SAMPLE_NAME}"
echo "JOB_PATH: ${JOB_PATH}"
echo "IN_PATH: ${IN_PATH}"
echo "OUT_PATH: ${OUT_PATH}"
echo "SINGLETONS: ${SINGLETONS}"
echo "PAIRED1: ${PAIRED1}"
echo "PAIRED2: ${PAIRED2}"



# create directories in scratch-dir
rm -rf ${JOB_PATH}
mkdir -p ${JOB_PATH}


# move fasta files to scratch-dir
cp -a -v "${IN_PATH}${SINGLETONS}" ${JOB_PATH}
cp -a -v "${IN_PATH}${PAIRED1}" ${JOB_PATH}
cp -a -v "${IN_PATH}${PAIRED2}" ${JOB_PATH}

echo "content of job dir: $(ls ${JOB_PATH})"


# move to job directory
cd ${JOB_PATH}

# run abyss assembler
abyss-pe \
name=${SAMPLE_NAME}\
j=32 \
k=${K} \
kc=${KC} \
B=6G \
```

```
v=-v \
in='${PAIRED1} ${PAIRED2}' \
se=${SINGLETONS}


echo "content of dir with results: $(ls ${JOB_PATH})"

# delete fasta-files from scratch dir after assembly
rm -rf "${JOB_PATH}${SINGLETONS}"
rm -rf "${JOB_PATH}${PAIRED1}"
rm -rf "${JOB_PATH}${PAIRED2}"


# copy output back to work dir
mkdir -p "${OUT_PATH}"
cp -a "${JOB_PATH}." "${OUT_PATH}"
rm -rf ${JOB_PATH}

conda deactivate
```

Old code...

Shell script to install libraries

```
# install libraries
echo 'archaea'
kraken2-build --download-library archaea --db
/work/mnikvell/data/Kraken2/dbs/PlusPFP-16_birds/

echo 'bacteria'
kraken2-build --download-library bacteria --db
/work/mnikvell/data/Kraken2/dbs/PlusPFP-16_birds/

echo 'plasmid'
kraken2-build --download-library plasmid --db
/work/mnikvell/data/Kraken2/dbs/PlusPFP-16_birds/

echo 'viral'
kraken2-build --download-library viral --db
/work/mnikvell/data/Kraken2/dbs/PlusPFP-16_birds/

echo 'human'
kraken2-build --download-library human --db
/work/mnikvell/data/Kraken2/dbs/PlusPFP-16_birds/

echo 'fungi'
kraken2-build --download-library fungi --db
/work/mnikvell/data/Kraken2/dbs/PlusPFP-16_birds/

echo 'plant'
kraken2-build --download-library plant --db
```

```
/work/mnikvell/data/Kraken2/dbs/PlusPFP-16_birds/

echo 'protozoa'
kraken2-build --download-library protozoa --db
/work/mnikvell/data/Kraken2/dbs/PlusPFP-16_birds/

echo 'UniVec_Core'
kraken2-build --download-library UniVec_Core --db
/work/mnikvell/data/Kraken2/dbs/PlusPFP-16_birds/


# add genomes (already downloaded) to library
echo 'genome chicken'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_016699485.1/GCA_01
6699485.1_bGalGal1.mat.broiler.GRCg7b_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}

echo 'genome great tit'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_001522545.3/GCA_00
1522545.3_Parus_major1.1_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}

echo 'genome blue tit'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_002901205.1/GCA_00
2901205.1_cyaCae2_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}

echo 'genome zebra finch'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_003957565.4/GCA_00
3957565.4_bTaeGut1.4.pri_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}

echo 'genome Tibetan ground-tit'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_000331425.1/GCA_00
0331425.1_PseHum1.0_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}

echo 'genome blood parasite'
kraken2-build --add-to-library
/work/mnikvell/data/genomes/genbank/protozoa/GCA_001625125.1/GCA_001625125.
1_ASM162512v1_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}
```

Shell script to build database on lido3-cluster

I also wrote a shell script to do all steps (including building the Bracken db) on the lido-cluster. But this takes a very long time. So maybe for the future I should do this in several smaller jobs (see

above).

```
#!/bin/bash -l
#SBATCH --partition=long
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=2-00:00:00
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=6G
#SBATCH --job-name=kraken_build_job
#SBATCH --mail-user=nikolas.vellnow@tu-dortmund.de
#SBATCH --mail-type=All

conda activate kraken

THREAD_NUM=32
DB_NAME=full_5_birds_with_bracken
DB_PATH=/scratch/mnikvell/kraken_job_${SLURM_JOBID}/${DB_NAME}/
OUT_PATH=/work/mnikvell/data/Kraken2/dbs/

echo "db name: ${DB_NAME}"
echo "db path: ${DB_PATH}"
echo "output path: ${OUT_PATH}"

# create directories in scratch-dir
rm -rf /scratch/mnikvell/kraken_job_${SLURM_JOBID}/
mkdir -p /scratch/mnikvell/kraken_job_${SLURM_JOBID}/
mkdir -p /scratch/mnikvell/kraken_job_${SLURM_JOBID}/${DB_NAME}


# scratch directory
echo "content of scratch dir: $(ls -R /scratch/mnikvell/)"

# move to job directory
cd /scratch/mnikvell/kraken_job_${SLURM_JOBID}/


# download taxonomy
kraken2-build --download-taxonomy --threads ${THREAD_NUM} --db "${DB_PATH}"

# download most dbs
echo 'archaea'
kraken2-build --download-library archaea --threads ${THREAD_NUM} --db
"${DB_PATH}"

echo 'bacteria'
kraken2-build --download-library bacteria --threads ${THREAD_NUM} --db
"${DB_PATH}"

echo 'plasmid'
kraken2-build --download-library plasmid --threads ${THREAD_NUM} --db
"${DB_PATH}"

echo 'viral'
```

```
kraken2-build --download-library viral --threads ${THREAD_NUM} --db
"${DB_PATH}"

echo 'human'
kraken2-build --download-library human --threads ${THREAD_NUM} --db
"${DB_PATH}"

echo 'fungi'
kraken2-build --download-library fungi --threads ${THREAD_NUM} --db
"${DB_PATH}"

echo 'plant'
kraken2-build --download-library plant --threads ${THREAD_NUM} --db
"${DB_PATH}"

echo 'protozoa'
kraken2-build --download-library protozoa --threads ${THREAD_NUM} --db
"${DB_PATH}"

echo 'UniVec_Core'
kraken2-build --download-library UniVec_Core --threads ${THREAD_NUM} --db
"${DB_PATH}"


# add genomes (already downloaded) to library
echo 'genome chicken'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_016699485.1/GCA_01
6699485.1_bGalGal1.mat.broiler.GRCg7b_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}

echo 'genome great tit'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_001522545.3/GCA_00
1522545.3_Parus_major1.1_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}

echo 'genome blue tit'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_002901205.1/GCA_00
2901205.1_cyaCae2_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}

echo 'genome zebra finch'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_003957565.4/GCA_00
3957565.4_bTaeGut1.4.pri_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}

echo 'genome Tibetan ground-tit'
kraken2-build -add-to-library
/work/mnikvell/data/genomes/genbank/vertebrate_other/GCA_000331425.1/GCA_00
0331425.1_PseHum1.0_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}
```

```
echo 'genome blood parasite'
kraken2-build --add-to-library
/work/mnikvell/data/genomes/genbank/protozoa/GCA_001625125.1/GCA_001625125.
1_ASM162512v1_genomic.fna \
--db "${DB_PATH}" --threads ${THREAD_NUM}


# build database
kraken2-build --build --db "${DB_PATH}" --threads ${THREAD_NUM}

# generate Bracken database file (databaseXmers.kmer_distrib)
bracken-build -d "${DB_PATH}" -t ${THREAD_NUM} -k 35 -l 150

# clean unnecessary files
kraken2-build --clean --db "${DB_PATH}" --threads ${THREAD_NUM}

# copy outputs back to
cp -a $DB_PATH $OUT_PATH

rm -rf /scratch/mnikvell/kraken_job_${SLURM_JOBID}/

conda deactivate
```