**Source code – Analysis of ChIP-Seq data**

#!/bin/bash

VER\_BOWTIE2="pgBowtie2.sh\_1.3.0"

#########################################

########## Description ##########

# functionality:

# reads list of fastq-files

# maps each fastq-file to the indicated genome, using Bowtie2

# writes bam-file as output

# sorts the output bam-file

# also writes a combined sorted bam-file, containing mapped & unmapped reads

#

# use:

# pgBowtie2.sh arg1 arg2 &>log.txt &; disown -h %1

#

# command line arguments:

# argument 1: one item per line

# row 1: full path to fastq directory

# row 2: full path to output-directory (bam- & sam-directory)

# row 3: full path to user directory; optional - if no directory is provided, the "output directory" will be used

# row 4: full path to bowtie2 genome

# row 5: additional tophat/bowtie2 parameters (optional)

# row 6: empty (\n)

# argument 2: list of fastq-files

# row 1 ff: one fastq-file name per row (root name, without extension '.fastq')

# last row: empty (\n)

# retrieve parameters

fastq\_dir=`sed -n '1p' < $1`

bam\_dir=`sed -n '2p' < $1`

user\_dir=`sed -n '3p' < $1`

genome=`sed -n '4p' < $1`

parameters=`sed -n '5p' < $1`

samples=$2

#map with tophat1/bowtie2

while read name; do

if [ -f "${fastq\_dir}/${name}.fastq" ]; then

printf "run tophat with the parameters: --no-coverage-search -p 40 $parameters for sample $name...\n"

tophat --no-coverage-search -g 1 -p 40 -o ${bam\_dir}/${name} $genome ${fastq\_dir}/${name}.fastq 2>${bam\_dir}/${name}\_tophat\_report.txt

fi

done < $samples

#sort and index bam file

printf "create sorted .bam files...\n"

while read name; do

if [ -f "${bam\_dir}/${name}/accepted\_hits.bam" ]; then

samtools sort -@ 35 ${bam\_dir}/${name}/accepted\_hits.bam -o ${bam\_dir}/${name}/${name}.mapped\_sort.bam

samtools index ${bam\_dir}/${name}/${name}.mapped\_sort.bam

samtools sort -@ 35 ${bam\_dir}/${name}/unmapped.bam -o ${bam\_dir}/${name}/unmapped\_sort.bam

samtools merge -@ 35 ${bam\_dir}/${name}\_sort.bam ${bam\_dir}/${name}/${name}.mapped\_sort.bam ${bam\_dir}/${name}/unmapped\_sort.bam

samtools index ${bam\_dir}/${name}\_sort.bam

# set extended attributes

old\_attributes=`getfattr -n user.comment ${fastq\_dir}/${name}.fastq | sed '1d' | sed 's/user.comment=//' | sed 's/\"//g'`

date=`date +%y%m%d`

added\_attributes="<MOD>$VER\_BOWTIE2</MOD><DAT>$date</DAT><GEN>$genome</GEN><FIL>${fastq\_dir}/${name}.fastq</FIL><PAR>$parameters</PAR>"

new\_attributes="${old\_attributes}${added\_attributes}"

setfattr -n user.comment -v "$new\_attributes" ${bam\_dir}/${name}\_sort.bam

setfattr -n user.comment -v "$new\_attributes" ${bam\_dir}/${name}\_sort.bam.bai

fi

done < $samples

printf "%s\n" $VER\_BOWTIE2 > $user\_dir/ver\_bowtie2#!/bin/bash

VER\_CHIPSEQ\_SPIKEIN\_MOUSE="pgChIPseq\_spikein\_mouse.sh\_1.2.0"

### NOTE: thread numbers reduced in "pgBowtie1.sh" & in lines 150, 151, 206, 207, 324, 370, 371 (03.12.2019)

########################################

########## description #########

# functionality:

# maps the fastq-files to the mm10 (bowtie1, for ungapped alignment)

# takes the unmapped reads (mm10): maps them to hg19

# takes the mapped reads (mm10): maps them to hg19

# establishes mapping stats

#

# command line arguments:

# argument 1: dir\_file

# one column only

# row 1: fastq\_dir = absolute path to directory with fastq-files; default is "$user\_dir/O1\_fastq"

# row 2: bam\_dir = absolute path to directory with bam-files; default is "$user\_dir/02\_bam"

# row 3: mapstat\_dir = absolute path to directory with mapstats-output; default is "$user\_dir/05\_stats"

# row 4: bg\_dir = absolute path to directory with bedgraph-output; default is "$user\_dir/03\_bedgraph"

#

# argument 2: info\_file

# row 1: bedgraph\_flag: if set to "G": bedgraph will be established

# row 2: experiment: name to be used for mapstats output

#

# argument 3: sample\_file

# col1: sample = root part of starting data file;

# - the part before the extension ".fastq"

#

# scripts used:

# pgBowtie1.sh

# pgMapStats.sh

##################################################################

########## define common data & read input data ##########

# dir\_file

fastq\_dir=`sed -n '1p'< $1`

bam\_dir=`sed -n '2p'< $1`

mapstat\_dir=`sed -n '3p'< $1`

bg\_dir=`sed -n '4p'< $1`

bedgraph\_flag=`sed -n '1p'< $2`

experiment=`sed -n '2p'< $2`

# sample\_file

sample\_file=$3

processed\_sample\_file=$sample\_file

# define some commen variables & paths

bowtie1\_dir="/media/user/data/\_genomes/bowtie1"

##############################################

########## start of the program ##########

start=`date +%s`

logname=`date +%y%m%d\_%H%M%S`

user\_dir=`pwd`

log\_long="${user\_dir}/log\_pgChIPseq\_spikein\_mouse\_${logname}\_ERROR.txt"

exec &>$log\_long

printf "%s\n%s %s\n\n" "start:" `date +%d/%m/%Y,%H:%M:%S` >> $log\_long

log\_short="${user\_dir}/log\_pgChIPseq\_spikein\_mouse\_${logname}\_short.txt"

printf "%s\n%s %s\n" $VER\_CHIPSEQ\_SPIKEIN\_MOUSE "start:" `date +%d/%m/%Y,%H:%M:%S` > $log\_short

contents\_dir\_file=`cat $1`

contents\_info\_file=`cat $2`

contents\_sample\_file=`cat $3`

printf "%s\n%s\n\n%s\n%s\n%s\n%s\n\n%s\n\n" "dir\_file:" "$contents\_dir\_file" "info\_file:" "$contents\_info\_file" "sample\_file:" "$contents\_sample\_file" >> $log\_short

############################################################

########## 1st call of pgBowtie1.sh: mm10 ##########

#

# command line arguments:

# argument 1: one item per line

# row 1: full path to fastq directory

# row 2: full path to output-directory (bam- & sam-directory)

# row 3: full path to user directory; optional - if no directory is provided, the "output directory" will be used

# row 4: full path to bowtie1 genome

# row 5: additional tophat/bowtie2 parameters (optional)

# row 6: empty (\n)

# argument 2: list of fastq-files

# row 1 ff: one fastq-file name per row (root name, without extension '.fastq')

# last row: empty (\n)

# define the command line variables

if [ ! -d "${bam\_dir}/mm10" ]; then

mkdir -p ${bam\_dir}/mm10

fi

printf "%s\n%s\n%s\n%s\n" "$fastq\_dir" "$bam\_dir/mm10" "$user\_dir" "$bowtie1\_dir/mm10" > ${user\_dir}/arg11\_bowtie1

awk '{OFS="\t"} {print $1}' $3 > ${user\_dir}/arg12\_bowtie1

# run the script

printf "%s\n\n" "pgChIPseq\_spikein\_mouse.sh:mm10" >> $log\_long

pgBowtie1.sh ${user\_dir}/arg11\_bowtie1 ${user\_dir}/arg12\_bowtie1

printf "%s\n" " " >> $log\_short

cat "$user\_dir/ver\_bowtie1" >> $log\_short

cat ${user\_dir}/arg1\*\_bowtie1 >> $log\_short

rm ${user\_dir}/arg1\*\_bowtie1

rm "$user\_dir/ver\_bowtie1"

##########################################################################

########## 1st call of pgMapstats.sh for 'mapped to mm10' ##########

#

# command line arguments:

# argument 1: one item per line

# row 1: full path to input directory (containing sorted bam-files with index)

# row 2: full path to output directory

# row 3: experiment name - to be used as the file name for output

# row 4: full path to user directory

# argument 2: sample\_file

# row 1 ff: full name of bam-file (including extension ".bam")

# last row: empty (\n)

if [ ! -d "$mapstat\_dir" ]; then

mkdir -p $mapstat\_dir # create the output-directory if it doesn't exist yet

fi

printf "%s\n%s\n%s\n%s\n" "${bam\_dir}/mm10" "$mapstat\_dir" "${experiment}\_mapped\_mm10" "$user\_dir" > ${user\_dir}/arg11\_mapstats

awk '{OFS="\t"} {print $1"\_sort.bam"}' $3 > ${user\_dir}/arg12\_mapstats

# run the script

printf "%s\n\n" "pgMapstats.sh mapped\_mm10" >> $log\_long

pgMapstats.sh arg11\_mapstats arg12\_mapstats &>>$log\_long

printf "%s\n\n" " " >> $log\_short

cat "$user\_dir/ver\_mapstats" >> $log\_short

cat ${user\_dir}/arg1\*\_mapstats >> $log\_short

rm ${user\_dir}/arg1\*\_mapstats

rm "$user\_dir/ver\_mapstats"

min=`cat "${mapstat\_dir}/${experiment}\_mapped\_mm10.min\_mapped\_reads.txt"`

printf "%s\t%s\n" "min\_mapped\_reads $genome:" $min >> $log\_short

####################################################################

##### establish fastq-files: mapped\_mm10 & unmapped\_mm10 #####

#

if [ ! -d "${bam\_dir}/mm10/unmapped\_mm10" ]; then

mkdir -p ${bam\_dir}/mm10/unmapped\_mm10

fi

if [ ! -d "${bam\_dir}/mm10/mapped\_mm10" ]; then

mkdir -p ${bam\_dir}/mm10/mapped\_mm10

fi

while read sample; do

samtools bam2fq -@ 5 -f 4 $bam\_dir/mm10/${sample}\_sort.bam > ${bam\_dir}/mm10/unmapped\_mm10/${sample}\_unmapped\_mm10.fastq # original: -@ 20

samtools bam2fq -@ 5 -F 4 $bam\_dir/mm10/${sample}\_sort.bam > ${bam\_dir}/mm10/mapped\_mm10/${sample}\_mapped\_mm10.fastq # original: -@ 20

# add extended attributes

old\_attributes=`getfattr -n user.comment $bam\_dir/mm10/${sample}\_sort.bam | sed '1d' | sed 's/user.comment=//' | sed 's/\"//g'`

date=`date +%y%m%d`

added\_attributes1="<MOD>$VER\_CHIPSEQ\_SPIKEIN\_MOUSE</MOD><DAT>$date</DAT><GEN>unmapped\_to\_mm10</GEN><FIL>$bam\_dir/mm10/${sample}\_sort.bam</FIL>"

added\_attributes2="<MOD>$VER\_CHIPSEQ\_SPIKEIN\_MOUSE</MOD><DAT>$date</DAT><GEN>mapped\_to\_mm10</GEN><FIL>$bam\_dir/mm10/${sample}\_sort.bam</FIL>"

new\_attributes1="${old\_attributes}${added\_attributes1}"

new\_attributes2="${old\_attributes}${added\_attributes2}"

setfattr -n user.comment -v "$new\_attributes1" ${bam\_dir}/mm10/unmapped\_mm10/${sample}\_unmapped\_mm10.fastq

setfattr -n user.comment -v "$new\_attributes2" ${bam\_dir}/mm10/mapped\_mm10/${sample}\_mapped\_mm10.fastq

done < $sample\_file

##################################################################################

########## 2nd call of pgBowtie1.sh: unmapped\_mm10 -> map to hg19 ##########

#

# command line arguments:

# argument 1: one item per line

# row 1: full path to fastq directory

# row 2: full path to output-directory (bam- & sam-directory)

# row 3: full path to user directory; optional - if no directory is provided, the "output directory" will be used

# row 4: full path to bowtie1 genome

# row 5: additional tophat/bowtie2 parameters (optional)

# argument 2: list of fastq-files

# row 1 ff: one fastq-file name per row (root name, without extension '.fastq')

# define the command line variables

if [ ! -d "${bam\_dir}/hg19/unmapped\_mm10" ]; then

mkdir -p ${bam\_dir}/hg19/unmapped\_mm10

fi

printf "%s\n%s\n%s\n%s\n" "${bam\_dir}/mm10/unmapped\_mm10" "${bam\_dir}/hg19/unmapped\_mm10" "$user\_dir" "$bowtie1\_dir/hg19" > ${user\_dir}/arg21\_bowtie1

awk '{OFS="\t"} {print $1"\_unmapped\_mm10"}' $3 > ${user\_dir}/arg22\_bowtie1

# run the script

printf "%s\n\n" "pgChIPseq\_spikein\_mouse.sh:hg19\_unmapped\_mm10" >> $log\_long

pgBowtie1.sh ${user\_dir}/arg21\_bowtie1 ${user\_dir}/arg22\_bowtie1

printf "%s\n" " " >> $log\_short

cat "$user\_dir/ver\_bowtie1" >> $log\_short

cat ${user\_dir}/arg2\*\_bowtie1 >> $log\_short

rm ${user\_dir}/arg2\*\_bowtie1

rm "$user\_dir/ver\_bowtie1"

###################################################################################

##### establish bam-files: mapped\_hg19\_only = unmapped\_mm10 -> mapped\_hg19 #####

#

if [ ! -d "${bam\_dir}/hg19/mapped\_hg19\_only" ]; then

mkdir -p ${bam\_dir}/hg19/mapped\_hg19\_only

fi

while read sample; do

samtools view -@ 5 -F 4 -O BAM -o ${bam\_dir}/hg19/mapped\_hg19\_only/${sample}\_only\_hg19.bam $bam\_dir/hg19/unmapped\_mm10/${sample}\_unmapped\_mm10\_sort.bam # original: -@ 20

samtools sort -@ 5 ${bam\_dir}/hg19/mapped\_hg19\_only/${sample}\_only\_hg19.bam -o ${bam\_dir}/hg19/mapped\_hg19\_only/${sample}\_only\_hg19\_sort.bam # original: -@ 20

rm ${bam\_dir}/hg19/mapped\_hg19\_only/${sample}\_only\_hg19.bam

samtools index ${bam\_dir}/hg19/mapped\_hg19\_only/${sample}\_only\_hg19\_sort.bam

mappedreads=`samtools idxstats ${bam\_dir}/hg19/mapped\_hg19\_only/${sample}\_only\_hg19\_sort.bam | grep -v "\*" | awk '{s+=$3} END {print s}'`

unmappedreads=`samtools idxstats ${bam\_dir}/hg19/mapped\_hg19\_only/${sample}\_only\_hg19\_sort.bam | grep "\*" | awk '{print $4}'`

total=`echo $((mappedreads+unmappedreads))`

# add extended attributes

old\_attributes=`getfattr -n user.comment $bam\_dir/hg19/mapped\_mm10/${sample}\_unmapped\_mm10\_sort.bam | sed '1d' | sed 's/user.comment=//' | sed 's/\"//g'`

date=`date +%y%m%d`

added\_attributes1="<MOD>$VER\_CHIPSEQ\_SPIKEIN\_MOUSE</MOD><GEN>only\_mapped\_to\_hg19</GEN><FIL>$bam\_dir/hg19/unmapped\_mm10/${sample}\_unmapped\_mm10\_sort.bam</FIL><NUM>$total</NUM><MAP>$mappedreads</MAP><UNM>$unmappedreads</UNM>"

new\_attributes1="${old\_attributes}${added\_attributes1}"

setfattr -n user.comment -v "$new\_attributes1" ${bam\_dir}/hg19/mapped\_hg19\_only/${sample}\_only\_hg19\_sort.bam

setfattr -n user.comment -v "$new\_attributes1" ${bam\_dir}/hg19/mapped\_hg19\_only/${sample}\_only\_hg19\_sort.bam.bai

done < $sample\_file

##############################################################################################

########## 2nd call of pgMapstats.sh: unmapped\_mm10 -> mapped to hg19 ##########

#

# command line arguments:

# argument 1: one item per line

# row 1: full path to input directory (containing sorted bam-files with index)

# row 2: full path to output directory

# row 3: experiment name - to be used as the file name for output

# row 4: full path to user directory

# argument 2: sample\_file

# row 1 ff: full name of bam-file (including extension ".bam")

# last row: empty (\n)

printf "%s\n%s\n%s\n%s\n" "${bam\_dir}/hg19/unmapped\_mm10" "$mapstat\_dir" "${experiment}\_unmapped\_mm10\_mapped\_hg19" "$user\_dir" > ${user\_dir}/arg21\_mapstats

awk '{OFS="\t"} {print $1"\_unmapped\_mm10\_sort.bam"}' $3 > ${user\_dir}/arg22\_mapstats

# run the script

printf "%s\n\n" "pgMapstats.sh mapped\_hg19\_unmapped\_mm10" >> $log\_long

pgMapstats.sh arg21\_mapstats arg22\_mapstats &>>$log\_long

printf "%s\n\n" " " >> $log\_short

cat "$user\_dir/ver\_mapstats" >> $log\_short

cat ${user\_dir}/arg2\*\_mapstats >> $log\_short

rm ${user\_dir}/arg2\*\_mapstats

rm "$user\_dir/ver\_mapstats"

min=`cat "${mapstat\_dir}/${experiment}\_unmapped\_mm10\_mapped\_hg19.min\_mapped\_reads.txt"`

printf "%s\t%s\n" "min\_mapped\_reads unmapped\_mm10\_mapped\_hg19:" $min >> $log\_short

##################################################################################

########## 3rd call of pgBowtie1.sh: mapped\_mm10 -> map to hg19 ############

#

# command line arguments:

# argument 1: one item per line

# row 1: full path to fastq directory

# row 2: full path to output-directory (bam- & sam-directory)

# row 3: full path to user directory; optional - if no directory is provided, the "output directory" will be used

# row 4: full path to bowtie1 genome

# row 5: additional tophat/bowtie2 parameters (optional)

# argument 2: list of fastq-files

# row 1 ff: one fastq-file name per row (root name, without extension '.fastq')

# define the command line variables

if [ ! -d "${bam\_dir}/hg19/mapped\_mm10" ]; then

mkdir -p ${bam\_dir}/hg19/mapped\_mm10

fi

printf "%s\n%s\n%s\n%s\n" "${bam\_dir}/mm10/mapped\_mm10" "${bam\_dir}/hg19/mapped\_mm10" "$user\_dir" "$bowtie1\_dir/hg19" > ${user\_dir}/arg31\_bowtie1

awk '{OFS="\t"} {print $1"\_mapped\_mm10"}' $3 > ${user\_dir}/arg32\_bowtie1

# run the script

printf "%s\n\n" "pgChIPseq\_spikein\_mouse.sh:hg19\_mapped\_mm10" >> $log\_long

pgBowtie1.sh ${user\_dir}/arg31\_bowtie1 ${user\_dir}/arg32\_bowtie1

printf "%s\n" " " >> $log\_short

cat "$user\_dir/ver\_bowtie1" >> $log\_short

cat ${user\_dir}/arg3\*\_bowtie1 >> $log\_short

rm ${user\_dir}/arg3\*\_bowtie1

rm "$user\_dir/ver\_bowtie1"

###########################################################################

########## 3rd call of pgMapstats.sh: mapped\_mm10 -> map to hg19 ####

#

# command line arguments:

# argument 1: one item per line

# row 1: full path to input directory (containing sorted bam-files with index)

# row 2: full path to output directory

# row 3: experiment name - to be used as the file name for output

# row 4: full path to user directory

# argument 2: sample\_file

# row 1 ff: full name of bam-file (including extension ".bam")

# last row: empty (\n)

printf "%s\n%s\n%s\n%s\n" "${bam\_dir}/hg19/mapped\_mm10" "$mapstat\_dir" "${experiment}\_mapped\_mm10\_mapped\_hg19" "$user\_dir" > ${user\_dir}/arg31\_mapstats

awk '{OFS="\t"} {print $1"\_mapped\_mm10\_sort.bam"}' $3 > ${user\_dir}/arg32\_mapstats

# run the script

printf "%s\n\n" "pgMapstats.sh mapped\_hg19\_mapped\_mm10" >> $log\_long

pgMapstats.sh arg31\_mapstats arg32\_mapstats &>>$log\_long

printf "%s\n\n" " " >> $log\_short

cat "$user\_dir/ver\_mapstats" >> $log\_short

cat ${user\_dir}/arg3\*\_mapstats >> $log\_short

rm ${user\_dir}/arg3\*\_mapstats

rm "$user\_dir/ver\_mapstats"

min=`cat "${mapstat\_dir}/${experiment}\_mapped\_mm10\_mapped\_hg19.min\_mapped\_reads.txt"`

printf "%s\t%s\n" "min\_mapped\_reads mapped\_mm10\_mapped\_hg19:" $min >> $log\_short

##################################################################################

##### establish fastq-files: mapped\_mm10 -> unmapped\_hg19 -> mapped\_mm10 #####

#

if [ ! -d "${bam\_dir}/mm10/unmapped\_hg19" ]; then

mkdir -p ${bam\_dir}/mm10/unmapped\_hg19

fi

while read sample; do

samtools bam2fq -@ 5 -f 4 $bam\_dir/hg19/mapped\_mm10/${sample}\_mapped\_mm10\_sort.bam > ${bam\_dir}/mm10/unmapped\_hg19/${sample}\_mapped\_mm10\_unmapped\_hg19.fastq # original: -@ 20

# add extended attributes

old\_attributes=`getfattr -n user.comment $bam\_dir/hg19/mapped\_mm10/${sample}\_mapped\_mm10\_sort.bam | sed '1d' | sed 's/user.comment=//' | sed 's/\"//g'`

date=`date +%y%m%d`

added\_attributes1="<MOD>$VER\_CHIPSEQ\_SPIKEIN\_MOUSE</MOD><DAT>$date</DAT><GEN>unmapped\_to\_mm10</GEN><FIL>$bam\_dir/hg19/mapped\_mm10/${sample}\_mapped\_mm10.bam</FIL>"

new\_attributes1="${old\_attributes}${added\_attributes1}"

setfattr -n user.comment -v "$new\_attributes1" ${bam\_dir}/mm10/unmapped\_hg19/${sample}\_mapped\_mm10\_unmapped\_hg19.fastq

done < $sample\_file

################################################################################################

########## 4th call of pgBowtie1.sh: mm10\_mapped -> hg19\_unmapped -> map to mm10 ##########

#

# command line arguments:

# argument 1: one item per line

# row 1: full path to fastq directory

# row 2: full path to output-directory (bam- & sam-directory)

# row 3: full path to user directory; optional - if no directory is provided, the "output directory" will be used

# row 4: full path to bowtie1 genome

# row 5: additional tophat/bowtie2 parameters (optional)

# argument 2: list of fastq-files

# row 1 ff: one fastq-file name per row (root name, without extension '.fastq')

printf "%s\n%s\n%s\n%s\n" "${bam\_dir}/mm10/unmapped\_hg19" "${bam\_dir}/mm10/unmapped\_hg19" "$user\_dir" "$bowtie1\_dir/mm10" > ${user\_dir}/arg41\_bowtie1

awk '{OFS="\t"} {print $1"\_mapped\_mm10\_unmapped\_hg19"}' $3 > ${user\_dir}/arg42\_bowtie1

# run the script

printf "%s\n\n" "pgChIPseq\_spikein\_mouse.sh:mm10\_mapped\_mm10\_unmapped\_hg19" >> $log\_long

pgBowtie1.sh ${user\_dir}/arg41\_bowtie1 ${user\_dir}/arg42\_bowtie1

printf "%s\n" " " >> $log\_short

cat "$user\_dir/ver\_bowtie1" >> $log\_short

cat ${user\_dir}/arg4\*\_bowtie1 >> $log\_short

rm ${user\_dir}/arg4\*\_bowtie1

rm "$user\_dir/ver\_bowtie1"

##################################################################################################

##### establish bam-files: mapped\_mm10\_only = mm10\_mapped -> hg19\_unmapped -> mm10\_mapped #####

#

if [ ! -d "${bam\_dir}/mm10/mapped\_mm10\_only" ]; then

mkdir -p ${bam\_dir}/mm10/mapped\_mm10\_only

fi

while read sample; do

samtools view -@ 5 -F 4 -O BAM -o ${bam\_dir}/mm10/mapped\_mm10\_only/${sample}\_only\_mm10.bam ${bam\_dir}/mm10/unmapped\_hg19/${sample}\_mapped\_mm10\_unmapped\_hg19\_sort.bam # original: -@ 20

samtools sort -@ 5 ${bam\_dir}/mm10/mapped\_mm10\_only/${sample}\_only\_mm10.bam -o ${bam\_dir}/mm10/mapped\_mm10\_only/${sample}\_only\_mm10\_sort.bam # original: -@ 20

rm ${bam\_dir}/mm10/mapped\_mm10\_only/${sample}\_only\_mm10.bam

samtools index ${bam\_dir}/mm10/mapped\_mm10\_only/${sample}\_only\_mm10\_sort.bam

mappedreads=`samtools idxstats ${bam\_dir}/mm10/mapped\_mm10\_only/${sample}\_only\_mm10\_sort.bam | grep -v "\*" | awk '{s+=$3} END {print s}'`

unmappedreads=`samtools idxstats ${bam\_dir}/mm10/mapped\_mm10\_only/${sample}\_only\_mm10\_sort.bam | grep "\*" | awk '{print $4}'`

total=`echo $((mappedreads+unmappedreads))`

# add extended attributes

old\_attributes=`getfattr -n user.comment ${bam\_dir}/mm10/unmapped\_hg19/${sample}\_mapped\_mm10\_unmapped\_hg19\_sort.bam | sed '1d' | sed 's/user.comment=//' | sed 's/\"//g'`

date=`date +%y%m%d`

added\_attributes1="<MOD>$VER\_CHIPSEQ\_SPIKEIN\_MOUSE</MOD><GEN>only\_mapped\_to\_hg19</GEN><FIL>${bam\_dir}/mm10/unmapped\_hg19/${sample}\_mapped\_mm10\_unmapped\_hg19\_sort.bam</FIL><NUM>$total</NUM><MAP>$mappedreads</MAP><UNM>$unmappedreads</UNM>"

new\_attributes1="${old\_attributes}${added\_attributes1}"

setfattr -n user.comment -v "$new\_attributes1" ${bam\_dir}/mm10/mapped\_mm10\_only/${sample}\_only\_mm10\_sort.bam

setfattr -n user.comment -v "$new\_attributes1" ${bam\_dir}/mm10/mapped\_mm10\_only/${sample}\_only\_mm10\_sort.bam.bai

done < $sample\_file

############################################################################################

########## 4th call of pgMapstats.sh: mm10\_mapped -> hg19\_unmapped -> map to mm10 ####

#

# command line arguments:

# argument 1: one item per line

# row 1: full path to input directory (containing sorted bam-files with index)

# row 2: full path to output directory

# row 3: experiment name - to be used as the file name for output

# row 4: full path to user directory

# argument 2: sample\_file

# row 1 ff: full name of bam-file (including extension ".bam")

# last row: empty (\n)

printf "%s\n%s\n%s\n%s\n" "${bam\_dir}/mm10/unmapped\_hg19" "$mapstat\_dir" "${experiment}\_mapped\_mm10\_unmapped\_hg19" "$user\_dir" > ${user\_dir}/arg41\_mapstats

awk '{OFS="\t"} {print $1"\_mapped\_mm10\_unmapped\_hg19\_sort.bam"}' $3 > ${user\_dir}/arg42\_mapstats

# run the script

printf "%s\n\n" "pgMapstats.sh mapped\_hg19\_mapped\_mm10" >> $log\_long

pgMapstats.sh arg41\_mapstats arg42\_mapstats &>>$log\_long

printf "%s\n\n" " " >> $log\_short

cat "$user\_dir/ver\_mapstats" >> $log\_short

cat ${user\_dir}/arg4\*\_mapstats >> $log\_short

rm ${user\_dir}/arg4\*\_mapstats

rm "$user\_dir/ver\_mapstats"

min=`cat "${mapstat\_dir}/${experiment}\_mapped\_mm10\_unmapped\_hg19.min\_mapped\_reads.txt"`

printf "%s\t%s\n" "min\_mapped\_reads mapped\_hg19\_mapped\_mm10:" $min >> $log\_short

###############################################

########## end of the program ##########

end=`date +%s`

runtime=$((end-start))

printf "\n%s %s\n" "end:" `date +%d/%m/%Y,%H:%M:%S` >> $log\_short

printf '%s %02dh:%02dm:%02ds\n' "Run duration: " $(($runtime/3600)) $(($runtime%3600/60)) $(($runtime%60)) >>$log\_short

#!/bin/bash

VER\_BEDGRAPH="pgBedgraph.sh\_1.2.0"

#########################################

########## Description ##########

# functionality:

# reads list of bam-files

# randomly extracts indicated number of reads & writes corresponding output

# sorts the output bam-file

# writes bedgraph file for the shortened bam-file

#

# use:

# pgBedgraph.sh arg1 arg2 &>log.txt &; disown -h %1

#

# command line arguments:

# argument 1: one item per line

# row 1: full path to input directory (containing bam-files)

# row 2: full path to output directory (containing bedgraph-files)

# row 3: full path to user directory; optional - if no directory is provided, the "output directory" will be used

# row 4: genome length file (full path), containing in col1 the chromosome names & in col2 the number of nucleotids

# row 5: empty (\n)

# argument 2: sample\_list

# each row: one sample

# col 1 ff: full name of bam-file (without extension ".bam")

# col 2 ff: number of reads to be used

# last row: empty (\n)

# retrieve parameters

bam\_dir=`sed -n '1p' < $1`

bg\_dir=`sed -n '2p' < $1`

user\_dir=`sed -n '3p' < $1`

genome=`sed -n '4p' < $1`

samples=$2

# define sub for random number generation

get\_seeded\_random()

{

seed="$1"

openssl enc -aes-256-ctr -pass pass:"$seed" -nosalt </dev/zero 2>/dev/null

}

# do the analysis

while read name number; do

if [ -f "${bam\_dir}/${name}.bam" ]; then

printf "calculate bedgraph for sample $name ...\n"

samtools view -@ 20 -H ${bam\_dir}/${name}.bam > ${bam\_dir}/${name}.header.txt

samtools view -@ 20 -F 4 ${bam\_dir}/${name}.bam | grep -v '^@' - | shuf - --random-source=<(get\_seeded\_random 42) | head -n ${number} | cat ${bam\_dir}/${name}.header.txt - | samtools view -@ 20 -bS - |samtools sort -@ 20 - -o ${bam\_dir}/${name}\_random${number}mapped.bam

samtools index ${bam\_dir}/${name}\_random${number}mapped.bam

rm ${bam\_dir}/${name}.header.txt

bedtools genomecov -bg -g $genome -ibam ${bam\_dir}/${name}\_random${number}mapped.bam > ${bg\_dir}/${name}\_random${number}mapped.bedgraph

# set extended attributes

old\_attributes=`getfattr -n user.comment ${bam\_dir}/${name}.bam | sed '1d' | sed 's/user.comment=//' | sed 's/\"//g'`

date=`date +%y%m%d`

added\_attributes="<MOD>$VER\_BEDGRAPH</MOD><DAT>$date</DAT><FIL>${bam\_dir}/${name}.bam</FIL><GEN>$genome</GEN>"

new\_attributes="${old\_attributes}${added\_attributes}"

setfattr -n user.comment -v "$new\_attributes" ${bg\_dir}/${name}\_random${number}mapped.bedgraph

fi

done < $samples

printf "%s\n" $VER\_BEDGRAPH > $user\_dir/ver\_bedgraph