sequenza-utils Documentation

Release 2.2.0

Francesco Favero

Aug 20, 2019

Contents

Sequenza-utils is The supporting python library for the sequenza R package.

Sequenza is a software for the estimation and quantification of purity/ploidy and copy number alteration in sequencing experiments of tumor samples. Sequenza-utils provide command lines programs to transform common NGS file format - such as BAM, pileup and VCF - to input files for the R package

CHAPTER 1

User documentation

1.1 Installation

The sequenza-utils code is hosted in BitBucket.

Supported Python version are 2.7+, including python3, and pypy.

Sequenza-utils can be installed either via the Python Package Index (PyPI) or from the git repository.

The package uses the external command line tools samtools and tabix. For the package to function correctly such programs need to be installed in the system

1.1.1 Latest release via PyPI

To install the latest release via PyPI using pip:

```
pip install sequenza-utils
```

1.1.2 Development version

To use the test suite for the package is necessary to install also bwa Using the latest development version directly from the git repository:

```
git clone https://bitbucket.org/sequenzatools/sequenza-utils
cd sequenza-utils
python setup.py test
python setup.py install
```

1.2 Command line interface

The sequenza-utils comprehends several programs. All programs are accessible from the sequenza-utils command line interface.

```
$ sequenza-utils --help
usage: sequenza-utils [-h] [-v]
                      {bam2seqz,gc_wiggle,pileup2acgt,seqz_binning,seqz_merge,
↪snp2seqz}
                      ...

Sequenza Utils is a collection of tools primarily design to convert bam, pileup and␣
↪vcf files to seqz files, the format used in the sequenza R package

positional arguments:
    bam2seqz        Process a paired set of BAM/pileup files (tumor and
                    matching normal), and GC-content genome-wide
                    information, to extract the common positions withA and
                    B alleles frequencies
    gc_wiggle       Given a fasta file and a window size it computes the GC
                    percentage across the sequences, and returns a file in
                    the UCSC wiggle format.
    pileup2acgt     Parse the format from the samtools mpileup command, and
                    report the occurrence of the 4 nucleotides in each
                    position.
    seqz_binning    Perform the binning of the seqz file to reduce file
                    size and memory requirement for the analysis.
    seqz_merge      Merge two seqz files
    snp2seqz        Parse VCFs and other variant and coverage formats to
                    produce seqz files

optional arguments:
  -h, --help        show this help message and exit
  -v, --verbose     Show all logging information

This is version 2.1.9999b1 - Favero Francesco - 22 February 2019
```

1.2.1 CG wiggle

```
$ sequenza-utils gc_wiggle
error: argument -f/--fasta is required
usage: sequenza-utils gc_wiggle -f FASTA [-o OUT] [-w WINDOW]

optional arguments:
  -f FASTA, --fasta FASTA
                        the fasta file. It can be a file name or "-" to use
                        STDIN
  -o OUT                Output file "-" for STDOUT
  -w WINDOW             The window size to calculate the GC-content percentage
```

1.2.2 BAM/mpileup to seqz

```
$ sequenza-utils bam2seqz
error: argument -n/--normal is required
usage: sequenza-utils bam2seqz [-p] -n NORMAL -t TUMOR -gc GC [-F FASTA]
                               [-o OUT] [-n2 NORMAL2] [-C CHR [CHR ...]]
                               [--parallel NPROC] [-S SAMTOOLS] [-T TABIX]
                               [-q QLIMIT] [-f QFORMAT] [-N N] [--hom HOM]
                               [--het HET] [--het_f HET_F]

Input/Output:
  Input and output files.

  -p, --pileup          Use pileups as input files instead of BAMs.
  -n NORMAL, --normal NORMAL
                        Name of the BAM/pileup file from the reference/normal
                        sample
  -t TUMOR, --tumor TUMOR
                        Name of the BAM/pileup file from the tumor sample
  -gc GC                The GC-content wiggle file
  -F FASTA, --fasta FASTA
                        The reference FASTA file used to generate the
                        intermediate pileup. Required when input are BAM
  -o OUT, --output OUT  Name of the output file. To use gzip compression name
                        the file ending in .gz. Default STDOUT.
  -n2 NORMAL2, --normal2 NORMAL2
                        Optional BAM/pileup file used to compute the
                        depth.normal and depth-ratio, instead of using the
                        normal BAM.

Genotype:
  Options regarding the genotype filtering.

  --hom HOM             Threshold to select homozygous positions. Default 0.9.
  --het HET             Threshold to select heterozygous positions. Default
                        0.25.
  --het_f HET_F         Threshold of frequency in the forward strand to trust
                        heterozygous calls. Default -0.2 (Disabled, effective
                        with values >= 0).

Subset indexed files:
  Option regarding samtools and bam indexes.

  -C CHR [CHR ...], --chromosome CHR [CHR ...]
                        Argument to restrict the input/output to a chromosome
                        or a chromosome region. Coordinate format is Name:pos
                        .start-pos.end, eg: chr17:7565097-7590856, for a
                        particular region; eg: chr17, for the entire
                        chromosome. Chromosome names can checked in the
                        BAM/pileup files and are depending on the FASTA
                        reference used for alignment. Default behavior is to
                        not selecting any chromosome.
  --parallel NPROC      Defines the number of chromosomes to run in parallel.
                        The output will be divided in multiple files, one for
                        each chromosome. The file name will be composed by the
                        output argument (used as prefix) and a chromosome name
                        given by the chromosome argument list. This imply that
```

(continues on next page)

(continued from previous page)

```
                        both output and chromosome argument need to be set
                        correctly.
  -S SAMTOOLS, --samtools SAMTOOLS
                        Path of samtools exec file to access the indexes and
                        compute the pileups. Default "samtools"
  -T TABIX, --tabix TABIX
                        Path of the tabix binary. Default "tabix"

Quality and Format:
  Options that change the quality threshold and format.

  -q QLIMIT, --qlimit QLIMIT
                        Minimum nucleotide quality score for inclusion in the
                        counts. Default 20.
  -f QFORMAT, --qformat QFORMAT
                        Quality format, options are "sanger" or "illumina".
                        This will add an offset of 33 or 64 respectively to
                        the qlimit value. Default "sanger".
  -N N                  Threshold to filter positions by the sum of read depth
                        of the two samples. Default 20.
```

1.2.3 Binning seqz

```
$ sequenza-utils seqz_binning
error: argument -s/--seqz is required
usage: sequenza-utils seqz_binning -s SEQZ [-w WINDOW] [-o OUT] [-T TABIX]

optional arguments:
  -s SEQZ, --seqz SEQZ  A seqz file.
  -w WINDOW, --window WINDOW
                        Window size used for binning the original seqz file.
                        Default is 50.
  -o OUT                Output file "-" for STDOUT
  -T TABIX, --tabix TABIX
                        Path of the tabix binary. Default "tabix"
```

1.2.4 VCF to seqz

```
$ sequenza-utils snp2seqz
error: argument -v/--vcf is required
usage: sequenza-utils snp2seqz [-o OUTPUT] -v VCF -gc GC
                               [--vcf-depth VCF_DEPTH_TAG]
                               [--vcf-samples {n/t,t/n}]
                               [--vcf-alleles VCF_ALLELES_TAG]
                               [--preset {caveman,mutect,mpileup,strelka2_som}]
                               [--hom HOM] [--het HET] [--het_f HET_F] [-N N]
                               [-T TABIX]

Output:
  Output arguments

  -o OUTPUT, --output OUTPUT
```

(continues on next page)

(continued from previous page)

```
                        Output file. For gzip compressed output name the file
                        ending in .gz. Default STDOUT

Input:
  Input files

  -v VCF, --vcf VCF     VCF input file
  -gc GC                The GC-content wiggle file

VCF:
  Parsing option for the VCF file

  --vcf-depth VCF_DEPTH_TAG
                        Column separated VCF tags in the format column for the
                        read depth for the normal and for the tumor. Default
                        "DP:DP"
  --vcf-samples {n/t,t/n}
                        Order of the normal and tumor sample in the VCF
                        column, choices are "n/t" or "t/n". Default "n/t"
  --vcf-alleles VCF_ALLELES_TAG
                        Column separated VCF tags in the format column for the
                        alleles depth for the normal and for the tumor.
                        Default "AD:AD"
  --preset {caveman,mutect,mpileup,strelka2_som}
                        Preset set of options to parse VCF from popular
                        variant callers

Genotype:
  Genotype filtering options

  --hom HOM             Threshold to select homozygous positions. Default 0.9
  --het HET             Threshold to select heterozygous positions. Default
                        0.25.
  --het_f HET_F         Threshold of frequency in the forward strand to trust
                        heterozygous calls. Default -0.2 (Disabled, effective
                        with values >= 0).

Programs:
  Option to call and control externa programs

  -T TABIX, --tabix TABIX
                        Path of the tabix binary. Default "tabix"

Filters:
  Filter output file by various parameters

  -N N                  Threshold to filter positions by the sum of read depth
                        of the two samples. Default 20.
```

1.2.5 Merge overlapping seqz

```
$ sequenza-utils seqz_merge
error: argument -1/--seqz1 is required
usage: sequenza-utils seqz_merge [-o OUTPUT] -1 S1 -2 S2 [-T TABIX]
```

(continues on next page)

(continued from previous page)

```
Output:
  Output arguments

  -o OUTPUT, --output OUTPUT
                        Output file. For gzip compressed output name the file
                        ending in .gz. Default STDOUT

Input:
  Input files

  -1 S1, --seqz1 S1     First input file. If both input files contain the same
                        line, the information in the first file will be used
  -2 S2, --seqz2 S2     Second input file

Programs:
  Option to call and control externa programs

  -T TABIX, --tabix TABIX
                        Path of the tabix binary. Default "tabix"
```

1.3 User cookbook

In order to process BAM files and generate file index, the software samtools and tabix need to be installed in the system.

The package *sequenza-utils* includes several programs and it should support the generation of *seqz* files using commonly available input files, such as fasta, BAM and vcf files.

To write your own program using the *sequenza-utils* library, lease refer to the *API library interface*

1.3.1 Generate GC reference file

The GC content source required to generate *seqz* files must to be in the wiggle track format (WIG). In order to generate the wig file from any fasta file use the `gc_wiggle` program.

```
sequenza-utils gc_wiggle --fasta genome.fa.gz -w 50 -o genome_gc50.wig.gz
```

1.3.2 From BAM files

Normal and tumor BAM files

```
sequenza-utils bam2seqz --normal normal_sample.bam --tumor tumor_sample.bam \
    --fasta genome.fa.gz -gc genome_gc50.wig.gz --output sample.seqz.gz
```

Normal and tumor pileup files

```
sequenza-utils bam2seqz --normal normal_sample.pielup.gz \
    --tumor tumor_sample.pielup.gz --fasta genome.fa.gz \
    -gc genome_gc50.wig.gz --output sample.seqz.gz --pileup
```

Without normal, workaround

```
sequenza-utils bam2seqz --normal tumor_sample.bam --tumor tumor_sample.bam \
    --normal2 non_matching_normal_sample.bam --fasta genome.fa.gz \
    -gc genome_gc50.wig.gz --output sample.seqz.gz
```

1.3.3 Binning seqz, reduce memory

```
sequenza-utils seqz_binning --seqz sample.seqz.gz --window 50 \
    -o sample_bin50.seqz.gz
```

1.3.4 Seqz from VCF files

VCF files with DP and AD tags

```
sequenza-utils snp2seqz --vcf sample_calls.vcf.gz -gc genome_gc50.wig.gz \
    --output samples.seqz.gz
```

Mutect/Caveman/Strelka2 preset

```
sequenza-utils snp2seqz --vcf sample_calls.vcf.gz -gc genome_gc50.wig.gz \
    --preset mutect --output samples.seqz.gz
```

```
sequenza-utils snp2seqz --vcf sample_calls.vcf.gz -gc genome_gc50.wig.gz \
    --preset caveman --output samples.seqz.gz
```

```
sequenza-utils snp2seqz --vcf sample_calls.vcf.gz -gc genome_gc50.wig.gz \
    --preset strelka2_som --output samples.seqz.gz
```

1.3.5 Merge seqz files

Non overlapping calls (eg different chromosomes)

```
gzcat sample_chr1.seqz.gz sample_chr1.seqz.gz | \
    gawk '{if (NR!=1 && $1 != "chromosome") {print $0}}' | bgzip > \
    sample.seqz.gz
tabix -f -s 1 -b 2 -e 2 -S 1 sample.seqz.gz
```

Overlapping sample_calls

```
sequenza-utils seqz_merge --seqz1 sample_somatic.seqz.gz \
    --seqz2 sample_snps.seqz.gz --output samples.seqz.gz
```

CHAPTER 2

Library reference

2.1 API library interface

2.1.1 sequenza.izip

class sequenza.izip.**zip_coordinates**(*item1*, *item2*)
Merge two object that have coordinate chromosome/position. The format of the objects must be a tuple with (coordinates, data) where coordinate is a tuple with chromosome,position_start, position_end and data is a tuple with the data. The data of the two object will be merged for matching lines. For the first object only the start coordinate is taken into account.

sequenza.izip.**zip_fast**(*item1*, *item2*)
Use the native implementation of the heapq algorithm to sort and merge files chromosome-coordinate ordered. It assumes that the two files are position ordered and both files have the same chromosome order. It differs from zip_coordinates by the fact that this return all the position present in both files, group together the lines present in both

2.1.2 sequenza.wig

class sequenza.wig.**Wiggle**(*wig*)
Read/write wiggle files as iterable objects.

exception sequenza.wig.**WiggleError**(*message*)

2.1.3 sequenza.fasta

class sequenza.fasta.**Fasta**(*file*, *n=60*)
Creates an iterable with genomic coordinates from a fasta file

2.1.4 sequenza.pileup

sequenza.pileup.**acgt**(*pileup*, *quality*, *depth*, *reference*, *qlimit=53*, *noend=False*, *nostart=False*)
Parse the mpileup format and return the occurrence of each nucleotides in the given positions.

sequenza.pileup.**pileup_acgt**(*pileup*, *quality*, *depth*, *reference*, *qlimit=53*, *noend=False*, *nostart=False*)
Yet another version of the pileup parser. Used as a template for the C implementation, the old function still runs slightly faster, to my surprise...

2.1.5 sequenza.samtools

class sequenza.samtools.**bam_mpileup**(*bam*, *fasta*, *q=20*, *Q=20*, *samtools_bin='samtools'*, *regions=[]*)
Use samtools via subprocess and return an iterable object.

class sequenza.samtools.**indexed_pileup**(*pileup*, *tabix_bin='tabix'*, *regions=[]*)
Use tabix via subprocess to slice the pileup data and return an iterable object

sequenza.samtools.**program_version**(*program*)
Parse tabix or samtools help message in attempt to retrieve the software version: return format: [major, minor, *]

sequenza.samtools.**tabix_seqz**(*file_name*, *tabix_bin='tabix'*, *seq=1*, *begin=2*, *end=2*, *skip=1*)
Index a seqz file with tabix

2.1.6 sequenza.seqz

sequenza.seqz.**acgt_genotype**(*acgt_dict*, *freq_list*, *strand_list*, *hom_t*, *het_t*, *het_f*, *bases_list*)
Return the alleles in the genotype

sequenza.seqz.**unpack_data**(*data*)
Unpack normal, tumor and gc info from the specific touple structure and remove redundant information

2.1.7 sequenza.vcf

sequenza.vcf.**vcf_headline_content**(*line*)
Try to get the string enclosed by "< ... >" in the VCF header

sequenza.vcf.**vcf_parse**(*vcf_file, sample_order='n/t', field='FORMAT', depth=['DP', 'DP'], alleles=['AD', 'AD'], preset=None*)
Parse the specified tags of a vcf file to retrieve total and per-allele depth information.

Python Module Index

S

Index