cbgen

Release 1.0.1

Danilo Horta

Aug 18, 2022

CONTENTS:

BGEN is a file format for storing large genetic datasets. It supports both unphased genotypes and phased haplotype data with variable ploidy and number of alleles. It was designed to provide a compact data representation without sacrificing variant access performance. This Python package is a wrapper around the bgen library, a low-memory footprint reader that efficiently reads bgen files. It fully supports the bgen format specifications: 1.2 and 1.3; as well as their optional compressed formats.

CHAPTER

ONE

INSTALLATION

```
pip install cbgen
```

CHAPTER

TWO

USAGE EXAMPLE

```
>>> import cbgen
>>>
>>> bgen = cbgen.bgen_file(cbgen.example.get("haplotypes.bgen"))
>>> bgen.create_metafile("haplotypes.bgen.metafile")
>>> mf = cbgen.bgen_metafile("haplotypes.bgen.metafile")
>>> print(mf.npartitions)
1
>>> print(mf.nvariants)
4
>>> print(mf.partition_size)
4
>>> part = mf.read_partition(0)
>>> gt = bgen.read_genotype(part.variants.offset[0])
>>> print(gt.probability)
[[1. 0. 1. 0.]
 [0. 1. 1. 0.]
 [1. 0. 0. 1.]
 [0. 1. 0. 1.]]
>>> mf.close()
>>> bgen.close()
```

2.1 bgen_file

bgen_file(filepath)	BGEN file handler.
bgen_file.close()	Close file stream.
bgen_file.contain_samples	Check if it contains samples.
bgen_file.create_metafile(filepath[, verbose])	Create metafile file.
bgen_file.filepath	File path.
bgen_file.nsamples	Number of samples.
bgen_file.nvariants	Number of variants.
bgen_file.read_genotype(offset[, precision])	Read genotype.
bgen_file.read_probability(offset[, precision])	Read genotype probability.
bgen_file.read_samples()	Read samples.

class cbgen.**bgen_file**(*filepath*)

BGEN file handler.

```
>>> import cbgen
>>>
>>> bgen = cbgen.bgen_file(cbgen.example.get("haplotypes.bgen"))
>>> print(bgen.nvariants)
4
>>> print(bgen.nsamples)
4
>>> print(bgen.contain_samples)
True
>>> print(bgen.read_samples())
[b'sample_0' b'sample_1' b'sample_2' b'sample_3']
>>> mf = cbgen.bgen_metafile(cbgen.example.get("haplotypes.bgen.metafile"))
>>> part = mf.read_partition(0)
>>> gt = bgen.read_genotype(part.variants.offset[0])
>>> print(gt.probability)
[[1. 0. 1. 0.]
 [0. 1. 1. 0.]
 [1. 0. 0. 1.]
 [0. 1. 0. 1.]]
>>> mf.close()
>>> bgen.close()
```

Use *with*-statement context to guarantee file closing at the end.

```
>>> with cbgen.bgen_file(cbgen.example.get("haplotypes.bgen")) as bgen:
...     print(bgen.nvariants)
4
```

Parameters
filepath (Union[str, Path]) – BGEN file path.

close()
Close file stream.

property contain_samples: bool
Check if it contains samples.

Return type
bool

Returns
True if it does contain samples; False otherwise.

create_metafile(*filepath*, *verbose=False*)
Create metafile file.

Parameters
- **filepath** (Union[str, Path]) – File path.
- **verbose** – True to show progress; False otherwise (default).

property filepath: Path
File path.

Return type
Path

Returns
File path.

property nsamples: int

Number of samples.

Return type
int

Returns
Number of samples.

property nvariants: int

Number of variants.

Return type
int

Returns
Number of variants.

read_genotype(*offset*, *precision=64*)

Read genotype.

Parameters

- **offset** (int) – Variant offset.
- **precision** (int) – Probability precision in bits: 64 (default) or 32.

Return type
Genotype

Returns
Genotype.

Raises
RuntimeError – If invalid offset of or a file stream reading error occurs.

read_probability(*offset*, *precision=64*)

Read genotype probability.

Parameters

- **offset** (int) – Variant offset.
- **precision** (int) – Probability precision in bits: 64 (default) or 32.

Return type
Any

Returns
Probabilities.

Raises
RuntimeError – If invalid offset of or a file stream reading error occurs.

read_samples()

Read samples.

Return type
Any

Returns
Samples.

Raises
RuntimeError – If samples are not stored or a file stream reading error occurs.

2.2 bgen_metafile

bgen_metafile(filepath)	BGEN metafile file handler.
bgen_metafile.close()	Close file stream.
bgen_metafile.filepath	File path.
bgen_metafile.npartitions	Number of partitions.
bgen_metafile.nvariants	Number of variants.
bgen_metafile.partition_size	Number of variants per partition.
bgen_metafile.read_partition(index)	Read partition.

class cbgen.**bgen_metafile**(*filepath*)

BGEN metafile file handler.

```
>>> import cbgen
>>>
>>> bgen = cbgen.bgen_file(cbgen.example.get("haplotypes.bgen"))
>>> mf = cbgen.bgen_metafile(cbgen.example.get("haplotypes.bgen.metafile"))
>>> print(mf.npartitions)
1
>>> print(mf.nvariants)
4
>>> print(mf.partition_size)
4
>>> part = mf.read_partition(0)
>>> gt = bgen.read_genotype(part.variants.offset[0])
>>> print(gt.probability)
[[1. 0. 1. 0.]
 [0. 1. 1. 0.]
 [1. 0. 0. 1.]
 [0. 1. 0. 1.]]
>>> mf.close()
>>> bgen.close()
```

Use *with*-statement context to guarantee file closing at the end.

```
>>> with cbgen.bgen_metafile(cbgen.example.get("haplotypes.bgen.metafile")) as mf:
...     print(mf.npartitions)
1
```

Parameters
filepath (Union[str, Path]) – BGEN metafile file path.

Raises
RuntimeError – If a file stream reading error occurs.

close()

Close file stream.

property filepath: Path

File path.

Return type
Path

Returns
File path.

property npartitions: int

Number of partitions.

Return type
int

Returns
Number of partitions.

property nvariants: int

Number of variants.

Return type
int

Returns
Number of variants.

property partition_size: int

Number of variants per partition.

The last partition might have less variants than the partition size. Every other partition is guaranteed to have `partition_size` variants.

Return type
int

Returns
Partition size.

read_partition(*index*)

Read partition.

Parameters
index (int) – Partition index.

Return type
Partition

Returns
Partition.

Raises
RuntimeError – If index is invalid or a file stream reading error occurs.

2.3 cache_home

Downloaded example files are stored at the BGEN_CACHE_HOME folder.

```
>>> from cbgen import BGEN_CACHE_HOME
>>> BGEN_CACHE_HOME.is_dir()
True
```

2.4 example

cbgen.example.**get**(*filename*)

Get file path to an example.

Recognized file names:

- complex.23bits.no.samples.bgen
- haplotypes.bgen
- haplotypes.bgen.metadata.corrupted
- haplotypes.bgen.metafile
- wrong.metadata
- merged_487400x220000.bgen
- merged_487400x2420000.bgen
- merged_487400x4840000.bgen

Parameters
filename (str) – File name to fetch.

Return type
Path

Returns
File path.

2.5 typing

Support for type hints.

cbgen.typing.Genotype(probability, phased, ...)	Genotype.
cbgen.typing.Partition(offset, variants)	Partition of variants.
cbgen.typing.Variants(id, rsid, chromosome, ...)	Variants.

class cbgen.typing.**Genotype**(*probability*, *phased*, *ploidy*, *missing*)

Genotype.

```
>>> import cbgen
>>>
>>> bgen = cbgen.bgen_file(cbgen.example.get("haplotypes.bgen"))
>>> mf = cbgen.bgen_metafile(cbgen.example.get("haplotypes.bgen.metafile"))
>>> part = mf.read_partition(0)
>>> gt = bgen.read_genotype(part.variants.offset[0])
>>> print(type(gt))
<class 'cbgen.typing.Genotype'>
>>> print(gt.probability)
[[1. 0. 1. 0.]
 [0. 1. 1. 0.]
 [1. 0. 0. 1.]
 [0. 1. 0. 1.]]
>>> print(gt.phased)
True
>>> print(gt.ploidy)
[2 2 2 2]
>>> print(gt.missing)
[False False False False]
>>> mf.close()
>>> bgen.close()
```

Probability

Probability.

phased

Phasedness.

Type
Any

ploidy

Ploidy.

Type
Any

missing

Missingness.

Type
Any

class cbgen.typing.**Partition**(*offset*, *variants*)

Partition of variants.

```
>>> import cbgen
>>>
>>> bgen = cbgen.bgen_file(cbgen.example.get("haplotypes.bgen"))
>>> mf = cbgen.bgen_metafile(cbgen.example.get("haplotypes.bgen.metafile"))
>>> part = mf.read_partition(0)
>>> print(type(part))
<class 'cbgen.typing.Partition'>
>>> print(part.offset)
0
```

(continues on next page)

(continued from previous page)

```
>>> print(type(part.variants))
<class 'cbgen.typing.Variants'>
>>> mf.close()
>>> bgen.close()
```

offset

Partition offset.

Type
int

variants

Variants.

Type
cbgen.typing.Variants

class cbgen.typing.**Variants**(*id*, *rsid*, *chromosome*, *position*, *nalleles*, *allele_ids*, *offset*)

Variants.

```
>>> import cbgen
>>>
>>> bgen = cbgen.bgen_file(cbgen.example.get("haplotypes.bgen"))
>>> mf = cbgen.bgen_metafile(cbgen.example.get("haplotypes.bgen.metafile"))
>>> part = mf.read_partition(0)
>>> variants = part.variants
>>> print(type(variants))
<class 'cbgen.typing.Variants'>
>>> print(variants.size)
4
>>> print(variants.id[3])
b'SNP4'
>>> print(variants.rsid[3])
b'RS4'
>>> print(variants.chromosome[3])
b'1'
>>> print(variants.position[3])
4
>>> print(variants.nalleles[3])
2
>>> print(variants.allele_ids[3])
b'A,G'
>>> print(variants.offset[3])
273
>>> mf.close()
>>> bgen.close()
```

id

Identification.

Type
Any

rsid

Reference SNP cluster ID.

Type
Any

`chromosome`

Chromosome.

Type
Any

`position`

Position.

Type
Any

`nalleles`

Number of alleles per variant.

Type
Any

`allele_ids`

Allele identifications.

Type
Any

`offset`

Variant offset.

Type
Any

`property size: int`

Number of variants.

Return type
`int`

Returns
Number of variants.

CHAPTER

THREE

COMMENTS AND BUGS

You can get the source code and open issues on Github.

INDEX