

齐云使用手册

目录

一、	软件概述	2
二、	安装部署	2
三、	使用手册	3
(一)	models – 查询可用模型	3
(二)	consensus – 一键生成一致性序列	3
(三)	polish – 修正组装序列	4
(四)	evaluate – 序列评估	5
(五)	hap_variant – 单倍体突变检测	5
(六)	consensus2vcf – 从序列检测突变	6
(七)	evaluate_vcf – 突变检测评估	7
(八)	datafilter – 数据筛选	7
(九)	assemble – 初步组装	8

一、 软件概述

齐云是一款 Consensus 工具软件，其主要基于深度学习算法，根据 reads 组装生成 consensus 序列或进行单倍体突变检测。用户版利用 singularity 容器化技术，可在大部分 Linux 操作系统下使用，无需额外部署。除了生成 consensus 序列和单倍体突变检测两大核心功能外，也包括数据筛选、初步组装、polish 组装、序列评估等工具功能，具体使用方法见下文。

软件文件夹中的主要内容为：

- qiyun 为主入口脚本；
- models 文件夹下为当前可用的模型文件，模型名称与 basecall 模型一致，模型名称与各体系的关系见下表：

齐云-研发版模型名称与对应体系					
生化体系	适配测序试剂盒	适配测序芯片	适配建库试剂盒	模型类型	模型名
K1M2 体系-PCB-Lipid	QDS V1.1	Qcell-384-P V1.1	QDL-E V1.1	快速模型	QDSv1.1_QCell384Pv1.1_QDLEv1.1_DBv4.1.3
				高精度模型	QDSv1.1_QCell384Pv1.1_QDLEv1.1_NTGv4.2.2
K2M1 体系-PCB-Lipid	QDS V2.0	Qcell-384-P V2.0	QDL-E V1.0	快速模型	QDSv2.0_QCell384Pv2.0_QDLEv1.0_DBv4.1.3
				高精度模型	QDSv2.0_QCell384Pv2.0_QDLEv1.0_NTGv4.1.3
K1M2 体系-Si-Polymer	QDS V1.1	Qcell-384 V1.1	QDL-E V1.1	快速模型	QDSv1.1_QCell384v1.1_QDLEv1.1_DBv5.1.0
				高精度模型	QDSv1.1_QCell384v1.1_QDLEv1.1_NTGv5.1.0
K1M2 体系-Si-Lipid	QDS V1.1	Qcell-384 V1.0	QDL-E V1.1	快速模型	QDSv1.1_QCell384v1.0_QDLEv1.1_DBv4.1.1
				高精度模型	QDSv1.1_QCell384v1.0_QDLEv1.1_NTGv4.2.0

- qiyun_release_v1.3.sif 为齐云用户版 v1.3 镜像文件，若缺少该文件，在安装好 singularity 后，可通过命令“singularity pull --arch amd64 qiyun_release_v1.3.sif library://qitantech/qiyun/qiyun:v1.3”进行下载；
- singularity_setup 文件夹下是 singularity 容器环境的安装部署文件，如已安装 singularity 容器引擎可直接删除。

二、 安装部署

在 singularity_setup 文件夹下运行 `sudo bash singularity_setup.sh` 即可安装 singularity 容器引擎，需要计算机联网并且用户具有 root 权限。如果计算机上已经安装了 singularity 容器引擎，则可直接在文件夹下运行 qiyun 命令，无需该安装步骤。

若希望在任何位置都可以使用 qiyun 命令，则需要将该入口脚本的位置添加至 PATH 环境变量。常用方法为：`echo "export PATH=\"/path/to/qiyun:$PATH\"" >> ~/.bashrc && source ~/.bashrc`

三、 使用手册

运行 `qiyun <COMMAND>` 命令即可使用齐云的各种功能，通过 `qiyun -h` 即可查看软件可用的子命令和对应的功能，各个子命令的详细说明见下文。

(一) models – 查询可用模型

运行 `qiyun models` 可以列出当前版本下所有可用的模型名称，对应 models 文件夹下的各模型文件名。

(二) consensus – 一键生成一致性序列

运行 `qiyun consensus` 可以直接从 reads 一键生成 consensus 序列，其中包括数据筛选、初步组装、修正组装序列全套流程，使用方式：

```
qiyun consensus [-h] [-t THREADS] [-g GENOME_SIZE] [-o OUT_DIR] reads model
```

其中各参数含义如下：

- `-h`：展示帮助文档；
- 位置参数 `input_reads`：输入 reads 路径，需要是 fastq 文件；
- 位置参数 `model`：生成 consensus 序列使用的模型名称，最好选择与输入 reads 匹配的模型；
- `-t THREADS`：程序运行时使用的并行线程数量，默认值为 1；
- `-g GENOME_SIZE`：基因组的预估大小，reads 长度过短时必需，可以为 4500, 3.5k, 5.5m；
- `-o OUT_DIR`：输出文件夹，默认为输入 reads 的文件夹。输出文件主要包括：
 - `filtered.fastq`：根据 Q 值和长度筛选后的 reads；

- assembly.fasta: 初步组装序列;
- consensus.fasta: 最终结果 consensus 序列文件;
- polished_region.bed: 进行 polish 的区域;
- qiyun_predict_*.log: 运行日志文件, 包含深度过低区域的提示;
- reads_depth_summary.txt: 深度统计描述文件;

应用举例:

- `qiyun consensus /path/to/reads.fastq QDSv2.0_QCell384Pv2.0_QDLEv1.0_DBv4.1.3 -t 8 -o /path/to/output/directory`

(三) polish – 修正组装序列

运行 `qiyun polish` 可以基于 reads 修正组装序列, 生成 consensus 序列, 使用方式:

`qiyun polish [-h] [-t THREADS] [-o OUT_DIR] reads/bam draft model`

其中各参数含义如下:

- `-h`: 展示帮助文档;
- 位置参数 `reads/bam`: 输入文件路径, 可以是 fastq 格式的 reads 文件, 也可以是 reads 比对到组装序列的 bam 比对文件;
- 位置参数 `draft`: 组装序列文件路径, 需要是 fasta 文件;
- 位置参数 `model`: 生成 consensus 序列使用的模型名称, 最好选择与输入 reads 匹配的模型;
- `-t THREADS`: 程序运行时使用的并行线程数量, 默认值为 1;
- `-o OUT_DIR`: 输出文件夹, 默认为输入组装序列文件所在的文件夹。输出文件主要包括:
 - consensus.fasta: 最终结果 consensus 序列文件;
 - polished_region.bed: 进行 polish 的区域;
 - qiyun_predict_*.log: 运行日志文件, 包含深度过低区域的提示;
 - reads_depth_summary.txt: 深度统计描述文件;

应用举例:

- `qiyun polish /path/to/reads.fastq /path/to/assembly.fasta \`
`QDSv2.0_QCell384Pv2.0_QDLEv1.0_DBv4.1.3 -t 8 -o /path/to/output/directory`
- `qiyun polish /path/to/reads_to_assembly.bam /path/to/assembly.fasta`
`QDSv2.0_QCell384Pv2.0_QDLEv1.0_DBv4.1.3 -t 8 -o /path/to/output/directory`

(四) evaluate – 序列评估

运行 `qiyun evaluate` 可以评估组装序列或 consensus 序列。会将序列按指定长度切分，之后与参考序列比对，统计各类错误的数量和准确率 Q 值的分布，使用方式：

```
qiyun evaluate [-h] [-c CHUNK_SIZE] [-t THREADS] [-o OUT_DIR] [-r REF] consensus
```

其中各参数含义如下：

- `-h`：展示帮助文档；
- 位置参数 `consensus`：consensus 序列文件路径，需要是 fasta 文件；
- `-r REF`：参考序列文件路径，需要是 fasta 文件，在提供 reference 的时候，会对一致性序列的错误进行统计评估，未提供 reference 的时候，会使用 `quast.py`，`busco` 和 `viralverify` 进行评估；
- `-c CHUNK_SIZE`：定长切分 consensus 序列的长度，默认值为 100000；
- `-t THREADS`：程序运行时使用的并行线程数量，默认值为 1；
- `-o OUT_DIR`：输出文件夹，默认为输入 consensus 序列文件所在的文件夹。在提供 reference 时候输出文件主要包括：
 - `consensus_evaluation.csv`：consensus 序列评估报告；在不提供 reference 时候，输出文件主要包括：
 - `quast`：由 `quast.py` 评估输出结果所在文件夹；
 - `busco`：由 `busco` 评估输出结果所在文件夹；
 - `verify`：由 `viralverify` 评估输出结果所在文件夹；

应用举例：

- `qiyun evaluate /path/to/consensus.fasta -r /path/to/reference.fasta -c 100000 -t 8 -o /path/to/output/directory`
- `qiyun evaluate /path/to/consensus.fasta -c 100000 -t 8 -o /path/to/output/directory`

(五) hap_variant – 单倍体突变检测

运行 `qiyun hap_variant` 可以检测单倍体突变，使用方式：

```
qiyun hap_variant [-h] [-t THREADS] [-r REGIONS] [-d DEPTH] [-o OUT_DIR] reads ref model
```

其中各参数含义如下：

- `-h`：展示帮助文档；
- 位置参数 `reads`：输入 reads 路径，需要是 fastq 文件；

- 位置参数 ref: 参考序列文件路径, 需要是 fasta 文件;
- 位置参数 model: 检测单倍体突变使用的模型名称, 最好选择与输入 reads 匹配的模型;
- -t THREADS: 程序运行时使用的并行线程数量, 默认值为 1;
- -r REGIONS: 指定检测突变的区域, 需要是 bed 文件。默认在全基因组范围内检测单倍体突变;
- -d DEPTH: 高置信区域的深度阈值, 深度超过该值的区域视作高置信区域, 默认值为 15;
- -o OUT_DIR: 输出文件夹, 默认为输入 reads 所在的文件夹。输出文件主要包括:
 - consensus.fasta: 根据比对结果生成的 consensus 序列;
 - confident_regions.bed: 超过深度阈值的高置信区域;
 - output.vcf: 单倍体突变检测结果;
 - qiyun_hap_variant_*.log: 运行日志文件, 包含深度过低区域的提示;

应用举例:

- `qiyun hap_variant /path/to/reads.fastq /path/to/reference.fasta \`
`QDSv2.0_QCell384Pv2.0_QDLEv1.0_Dbv4.1.3 -t 8 -d 20 -r /path/to/regions.bed \`
`-o /path/to/output/directory`

(六) consensus2vcf – 从序列检测突变

运行 `qiyun consensus2vcf` 可以检测 consensus 序列相对于参考序列的单倍体突变, 使用方式:

```
qiyun consensus2vcf [-t THREADS] [-r REGIONS] [-o OUT_DIR] consensus ref
```

其中各参数含义如下:

- -h: 展示帮助文档;
- 位置参数 consensus: consensus 序列文件路径, 需要是 fasta 文件;
- 位置参数 ref: 参考序列文件路径, 需要是 fasta 文件;
- -t THREADS: 程序运行时使用的并行线程数量, 默认值为 1;
- -r REGIONS: 指定检测突变的区域, 需要是 bed 文件。默认在全基因组范围内检测单倍体突变;
- -o OUT_DIR: 输出文件夹, 默认为输入 reads 所在的文件夹。输出文件主要包括:
 - output.vcf: 单倍体突变检测结果;

应用举例:

- `qiyun consensus2vcf /path/to/consensus.fasta /path/to/reference.fasta -t 8 \`
`-r /path/to/regions.bed -o /path/to/output/directory`

(七) evaluate_vcf – 突变检测评估

运行 `qiyun evaluate_vcf` 可以评估检测到的单倍体突变，使用方式：

```
qiyun evaluate_vcf [-t THREADS] [-r REGIONS] [-o OUT_DIR] truth_vcf query_vcf ref
```

其中各参数含义如下：

- `-h`：展示帮助文档；
- 位置参数 `truth_vcf`：真实突变记录文件，需要是 `vcf` 文件；
- 位置参数 `query_vcf`：待评估突变记录文件，需要是 `vcf` 文件；
- 位置参数 `ref`：参考序列文件路径，需要是 `fasta` 文件；
- `-t THREADS`：程序运行时使用的并行线程数量，默认值为 1；
- `-r REGIONS`：指定评估突变的区域，需要是 `bed` 文件。默认在全基因组范围内检测单倍体突变；
- `-o OUT_DIR`：输出文件夹，默认为输入 `reads` 所在的文件夹。输出文件主要包括：
 - `variant_call_evaluation.csv`：突变检测评估报告；

应用举例：

- `qiyun evaluate_vcf /path/to/truth.vcf /path/to/query.vcf /path/to/reference.fasta -t 8 \`
`-r /path/to/regions.bed -o /path/to/output/directory`

(八) datafilter – 数据筛选

运行 `qiyun datafilter` 可以在 Q 值和长度上对 `reads` 进行筛选，并下采样 `reads` 至指定深度，使

用方式：

```
qiyun datafilter [-h] [-l MIN_LEN] [-q MIN_Q] [-d DEPTH] [-g GENOME_SIZE]  
                [-o OUT_DIR] input_reads
```

其中各参数含义如下：

- `-h`：展示帮助文档；
- 位置参数 `input_reads`：要筛选的 `reads` 路径，需要是 `fastq` 文件；
- `-l MIN_LEN`：筛选数据的最短长度，默认值为 400；
- `-q MIN_Q`：筛选数据的最小 Q 值，默认为 8；
- `-d DEPTH`：将数据下采样到指定深度，默认值为 0，表示不进行下采样；
- `-g GENOME_SIZE`：数据对应的基因组长度，用于下采样时估算深度，可以使用 `kb`, `MB` 等单位，如 1000, 3.9kb, 4.5MB 等，在指定 `-d` 选项后必需。

- `-o OUT_DIR`: 输出文件夹，默认为输入 reads 的文件夹。输出文件主要包括：
 - `filtered.fastq`: 筛选并下采样后的 reads；
 - `filter_pars.json`: 执行该命令时使用的参数，json 格式；

应用举例：

- `qiyun datafilter /path/to/reads.fastq -l 400 -q 8 -o /path/to/output/directory`
- `qiyun datafilter /path/to/reads.fastq -d 60 -g 4.5MB -o /path/to/output/directory`

(九) assemble – 初步组装

运行 `qiyun assemble` 可以对 reads 进行初步组装。根据输入的读长选择不同的组装工具，长读长

reads 使用 flye，短读长 reads 使用 canu，使用方式：

```
qiyun assemble [-h] [-n N_POL] [-g GENOME_SIZE] [-t THREADS] [-o OUT_DIR] input_reads
```

其中各参数含义如下：

- `-h`: 展示帮助文档；
- 位置参数 `input_reads`: 输入 reads 路径，需要是 fastq 文件；
- `-n N_POL`: 用于长读长 reads 组装，flye polish 轮数，默认值为 1；
- `-g GENOME_SIZE`: 短读长 reads 组装时必需，基因组的预估大小，可以为 4500, 3.5k, 5.5m；
- `-t THREADS`: 程序运行时使用的并行线程数量，默认值为 1；
- `-o OUT_DIR`: 输出文件夹，默认为输入 reads 的文件夹。输出文件包括：
 - `assembly.fasta`: 组装结果；

应用举例：

- `qiyun assemble /path/to/reads.fastq -n 1 -t 8 -o /path/to/output/directory`
- `qiyun assemble /path/to/reads.fastq -g 3.5k -t 8 -o /path/to/output/directory`