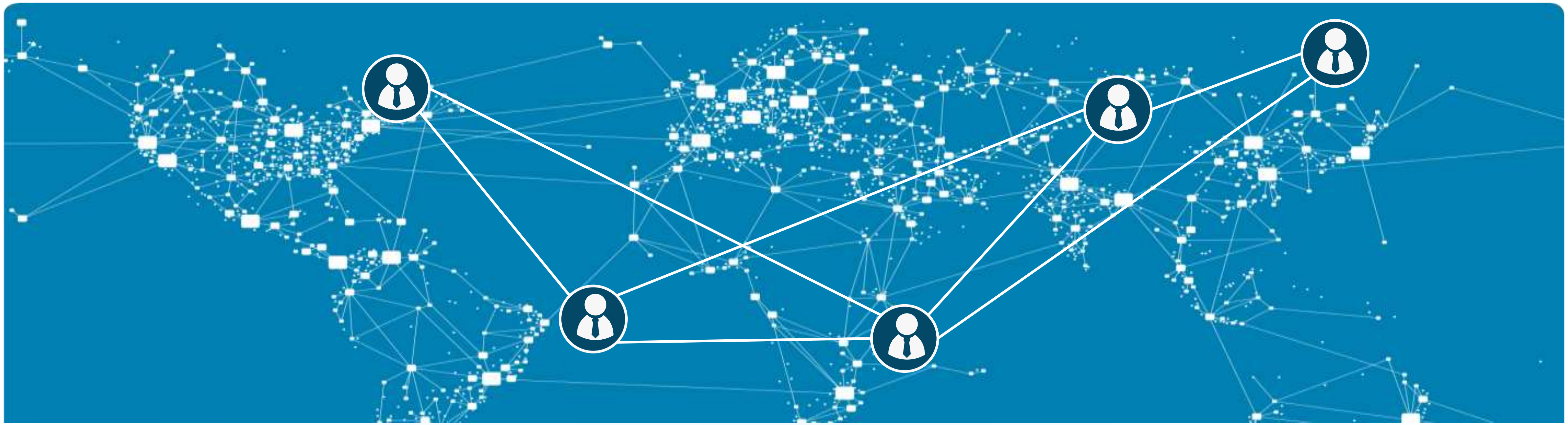
微生物组—宏基因组分析专题研讨会第24期



三代纳米孔宏基因组数据分析

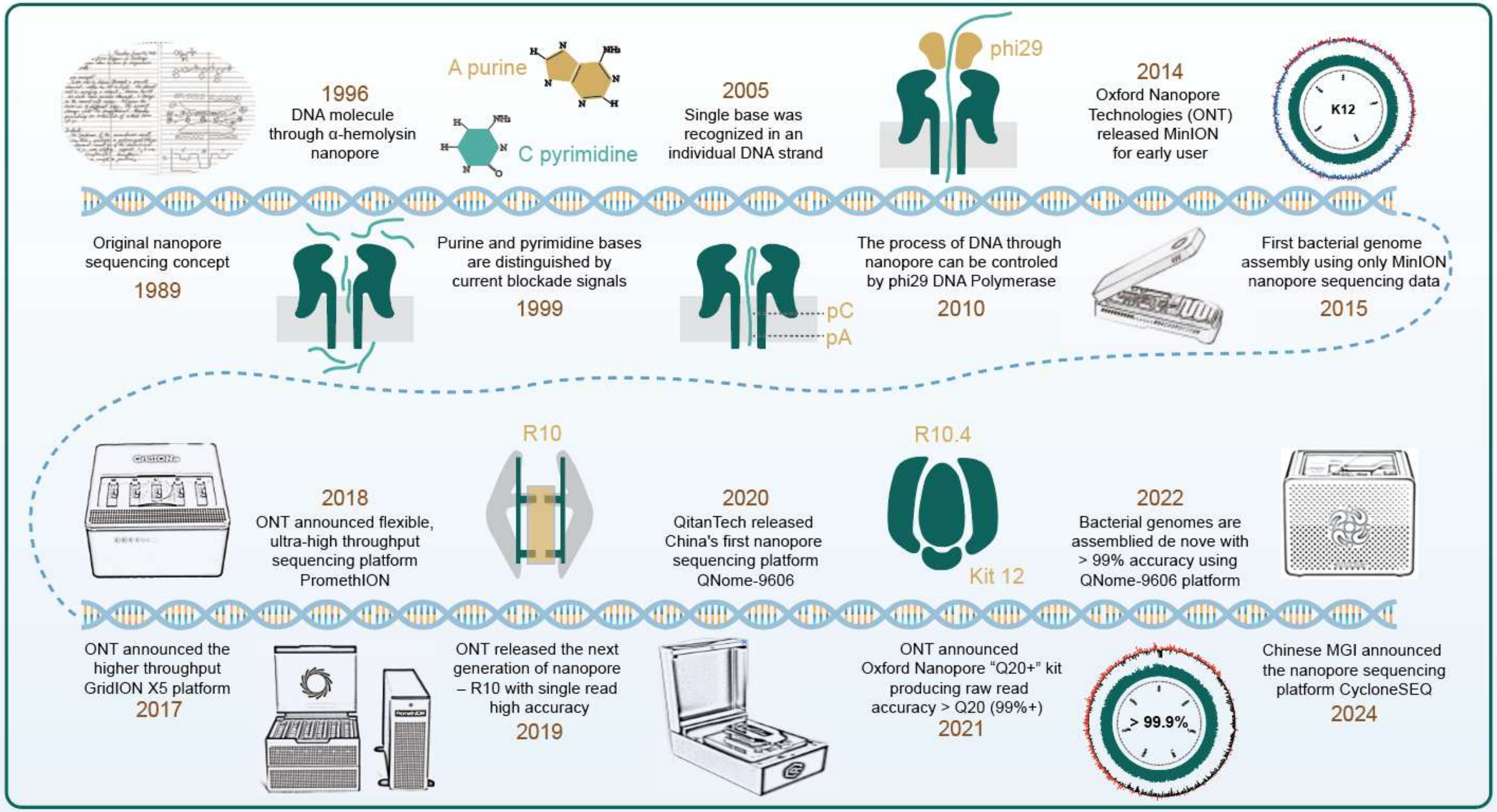
彭凯
2025年11月30日

课程目录

1. 简介：纳米孔测序与宏基因组

纳米孔测序技术概况

纳米孔测序技术的发展历史



Original nanopore sequencing concept
1989

1996
DNA molecule through α-hemolysin nanopore

Purine and pyrimidine bases are distinguished by current blockade signals
1999

2005
Single base was recognized in an individual DNA strand

The process of DNA through nanopore can be controled by phi29 DNA Polymerase
2010

2014
Oxford Nanopore Technologies (ONT) released MinION for early user

First bacterial genome assembly using only MinION nanopore sequencing data
2015

ONT announced the higher throughput GridION X5 platform
2017

2018
ONT announced flexible, ultra-high throughput sequencing platform PromethION

ONT released the next generation of nanopore – R10 with single read high accuracy
2019

2020
QitanTech released China's first nanopore sequencing platform QNome-9606

ONT announced Oxford Nanopore "Q20+" kit producing raw read accuracy > Q20 (99%+)
2021

2022
Bacterial genomes are assemblied de nove with > 99% accuracy using QNome-9606 platform

Chinese MGI announced the nanopore sequencing platform CycloneSEQ
2024

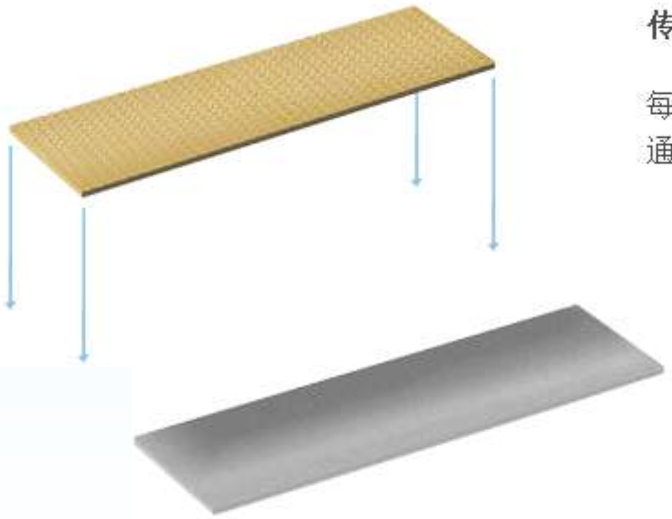
纳米孔测序芯片的结构

纳米孔

将一个蛋白质纳米孔设置于具电阻性的聚合物膜中。

微支架阵列

每个微支架可支撑一个膜和嵌入的纳米孔。阵列可使多个纳米孔在运输和使用过程中保持稳定。



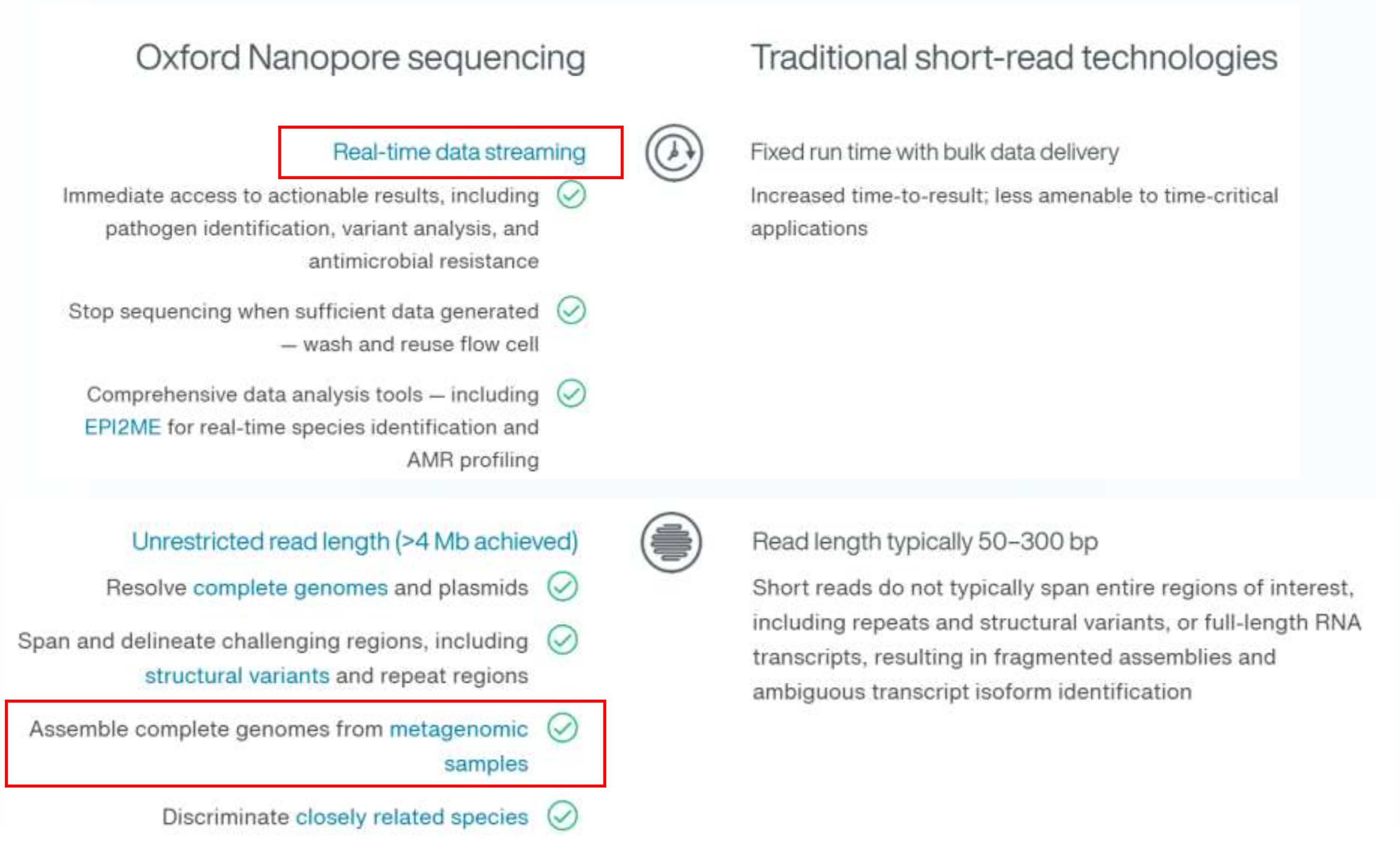
传感器芯片

每个微支架与其自身电极相对应，该电极连接至传感器阵列芯片中的通道上。传感器阵列可使用任何数量的通道来制造。

专用集成电路（ASIC）

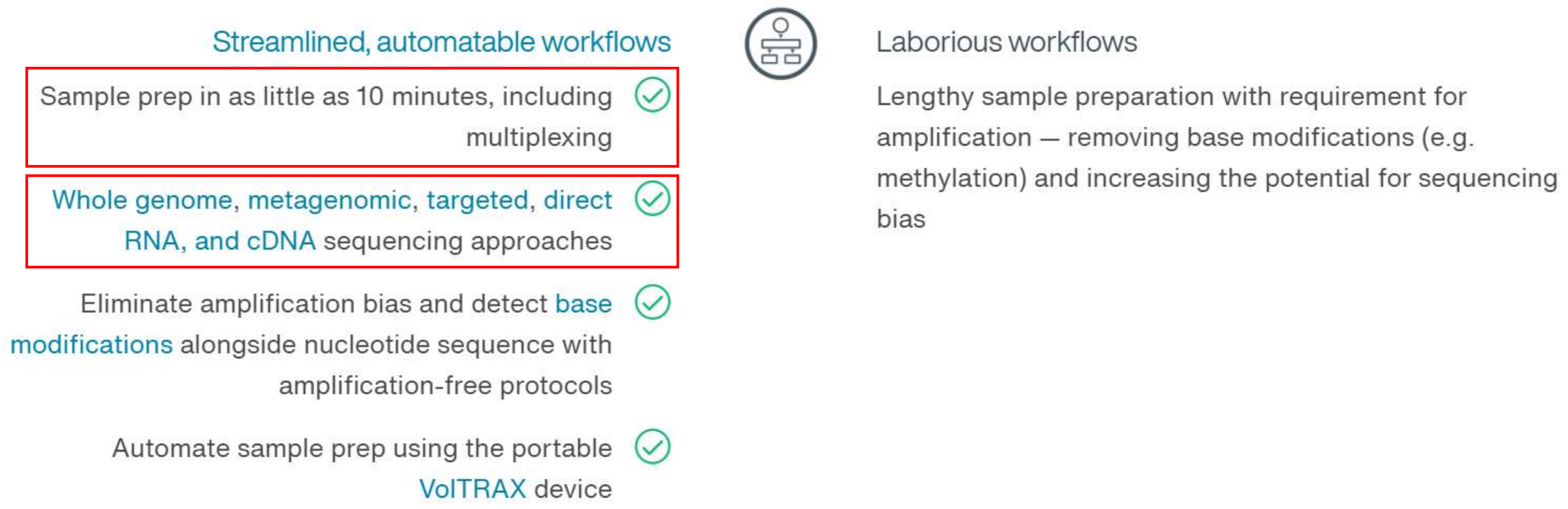
每个纳米孔通道均由定制的专用集成电路单独控制和测量。支持同时进行多个纳米孔实验。一个设备中可能包括多个专用集成电路，Oxford Nanopore 正在开发不同大小的专用集成电路，以用于不同目的。

纳米孔测序的优势



Oxford Nanopore sequencing	Traditional short-read technologies
Real-time data streaming	Fixed run time with bulk data delivery
Immediate access to actionable results, including pathogen identification, variant analysis, and antimicrobial resistance ✓	Increased time-to-result; less amenable to time-critical applications
Stop sequencing when sufficient data generated — wash and reuse flow cell ✓	
Comprehensive data analysis tools — including EPI2ME for real-time species identification and AMR profiling ✓	
Unrestricted read length (>4 Mb achieved)	Read length typically 50–300 bp
Resolve complete genomes and plasmids ✓	Short reads do not typically span entire regions of interest, including repeats and structural variants, or full-length RNA transcripts, resulting in fragmented assemblies and ambiguous transcript isoform identification
Span and delineate challenging regions, including structural variants and repeat regions ✓	
Assemble complete genomes from metagenomic samples ✓	
Discriminate closely related species ✓	

纳米孔测序的优势



Streamlined, automatable workflows

Sample prep in as little as 10 minutes, including multiplexing ✓

Whole genome, metagenomic, targeted, direct RNA, and cDNA sequencing approaches ✓

Eliminate amplification bias and detect base modifications alongside nucleotide sequence with amplification-free protocols ✓

Automate sample prep using the portable VolTRAX device ✓

Laborious workflows

Lengthy sample preparation with requirement for amplification — removing base modifications (e.g. methylation) and increasing the potential for sequencing bias

纳米孔测序的优势

1. 长读长：Reads可达Mb；

2. 设备成本低：测序芯片可清洗再生，重复利用；

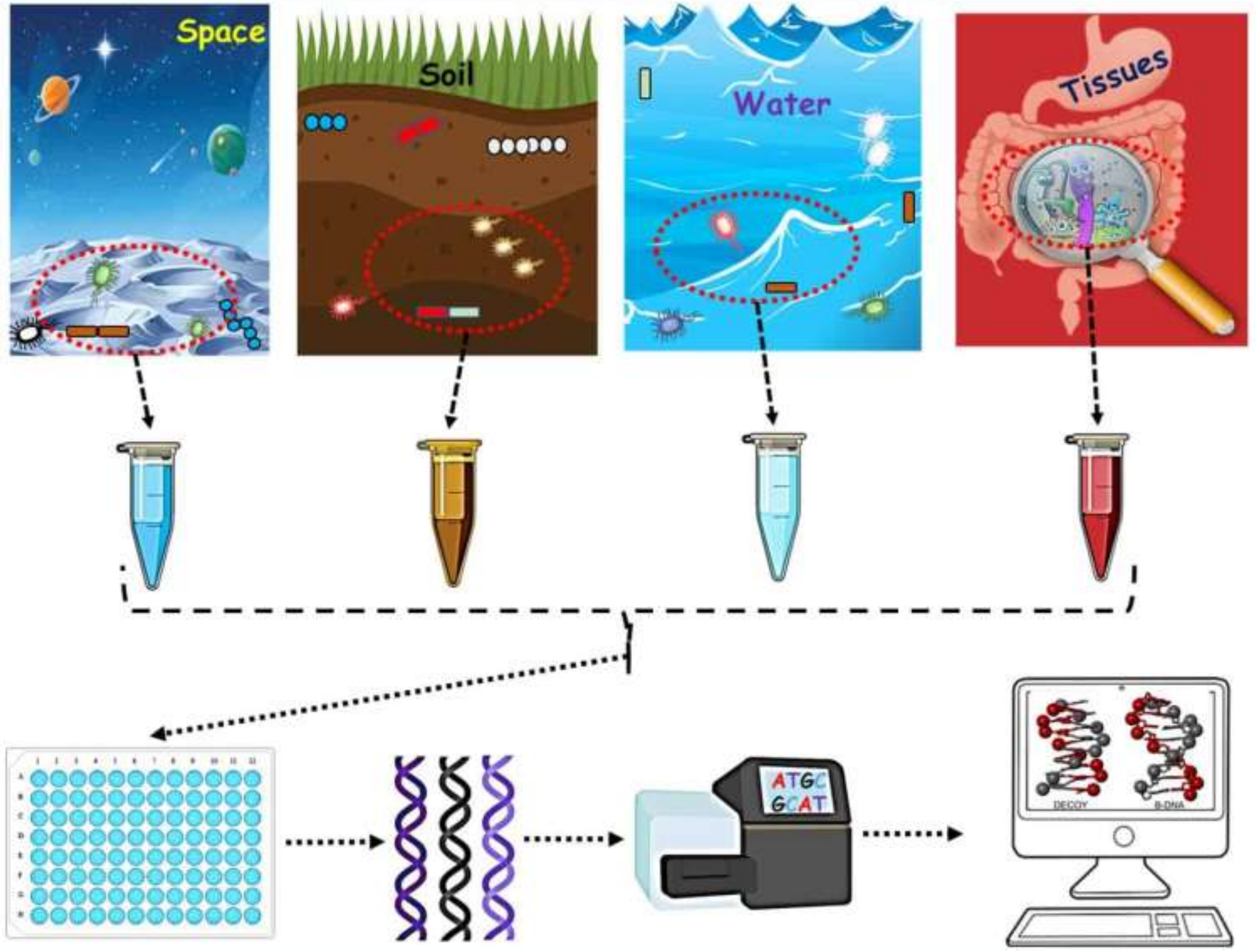
3. 实时获得序列信息：最快可在1小时内完成测序流程及数据分析，满足动态检测宏基因组需求；

4. 便携式测序装置：重量轻且占用空间小，可以随身携带随时测序；

5. 直接测序：直接测序原始DNA和RNA，不需要进行PCR扩增，避免了扩增偏好性；保留了原始碱基修饰信息，能够直接读出甲基化的胞嘧啶。

宏基因组研究概况

- 直接研究不同生态位中的微生物组DNA信息，对其进行测序，获取序列信息，再进行生物信息学分析



宏基因组研究对象

动物、人肠道基因集的构建

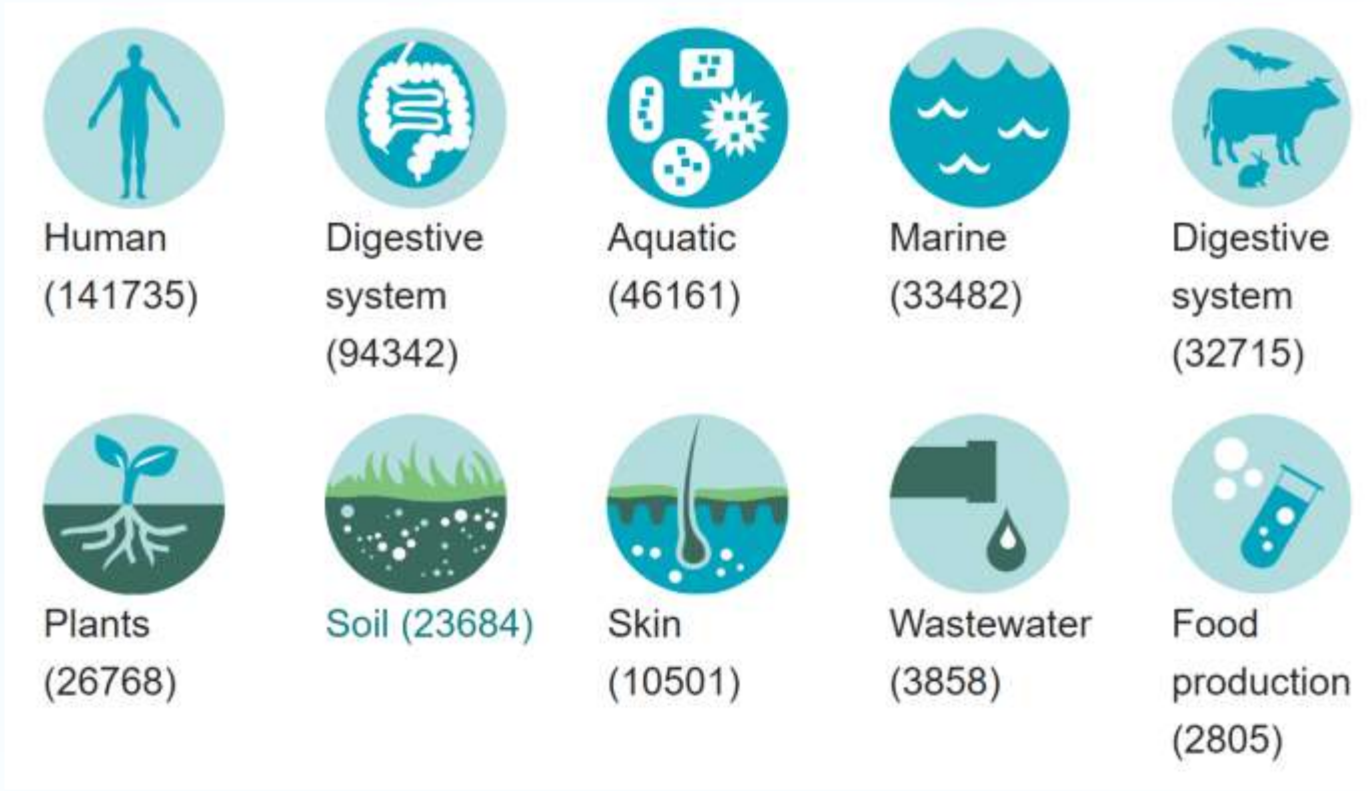
肠道微生物组多样性分析

新的功能基因的发掘，新物种的鉴定

肠道微生物组与宿主特定疾病的关系

饮食-肠道微生物组-宿主健康的关联

病原菌快速检测



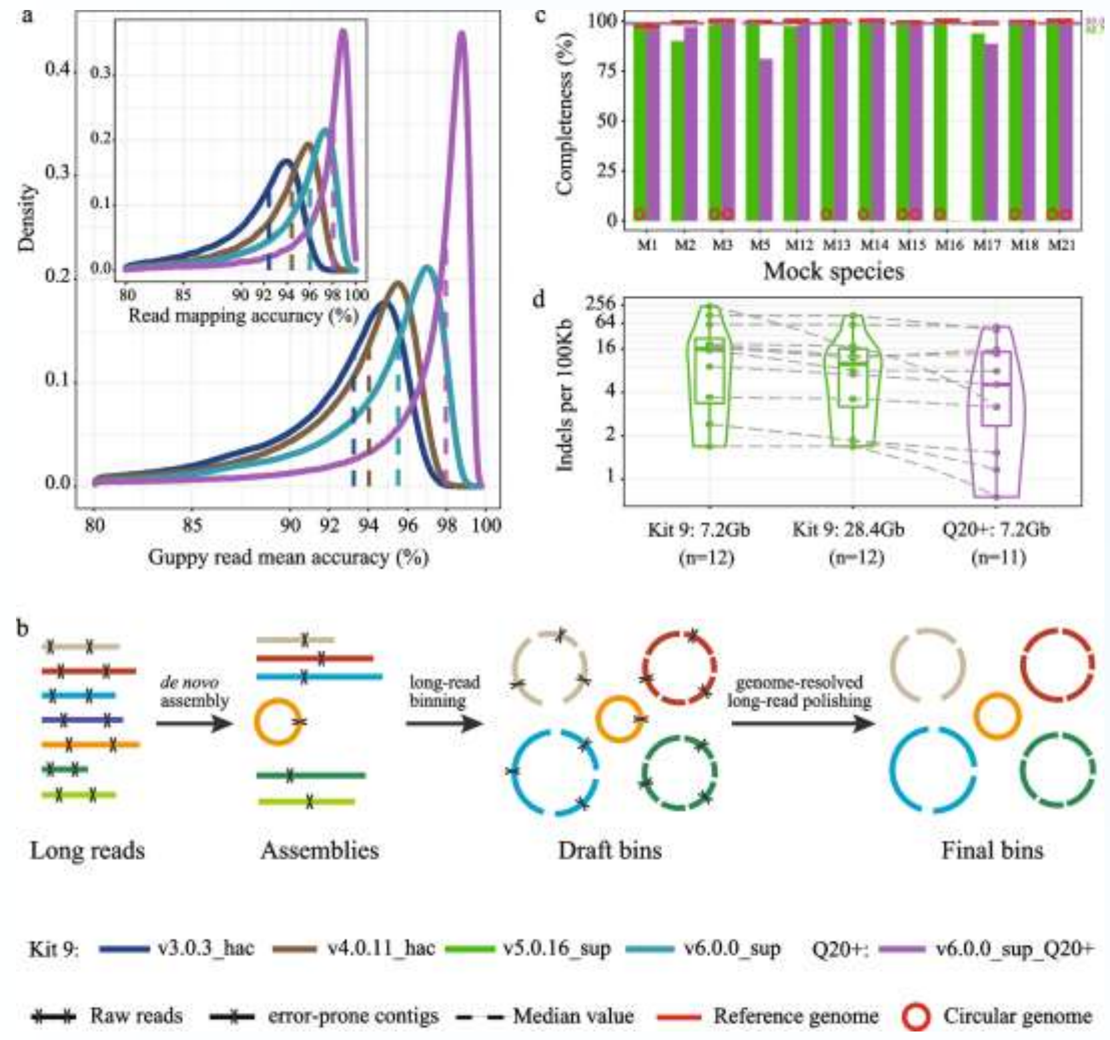
https://www.ebi.ac.uk/metagenomics

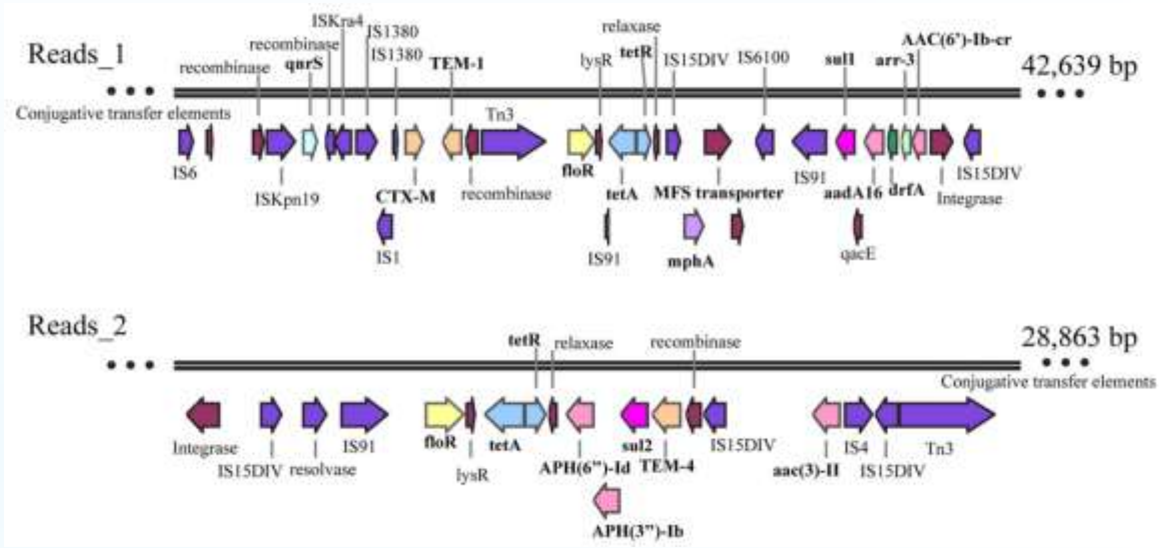
纳米孔宏基因组研究的发展历史

几个重要事件节点：

2014年：牛津纳米孔公司发布minion测序仪，纳米孔测序时代来临；
2016年：纳米孔宏基因组测序被用于快速病原检测；
2018年：港大张彤教授首次将纳米孔宏基因组测序应用到环境耐药组研究中，并建立相应研究方法；
2020年：利用纳米孔宏基因组测序可以从人肠道中重建几乎完整的细菌基因组；
2023年：超深度纳米孔长读宏基因组揭示人肠道微生物组中低丰度物种的基因组和功能特征；
2025年：NCBI的SRA数据库已经有超过4万条不同的纳米孔宏基因组测序数据集……

纳米孔宏基因组的优势



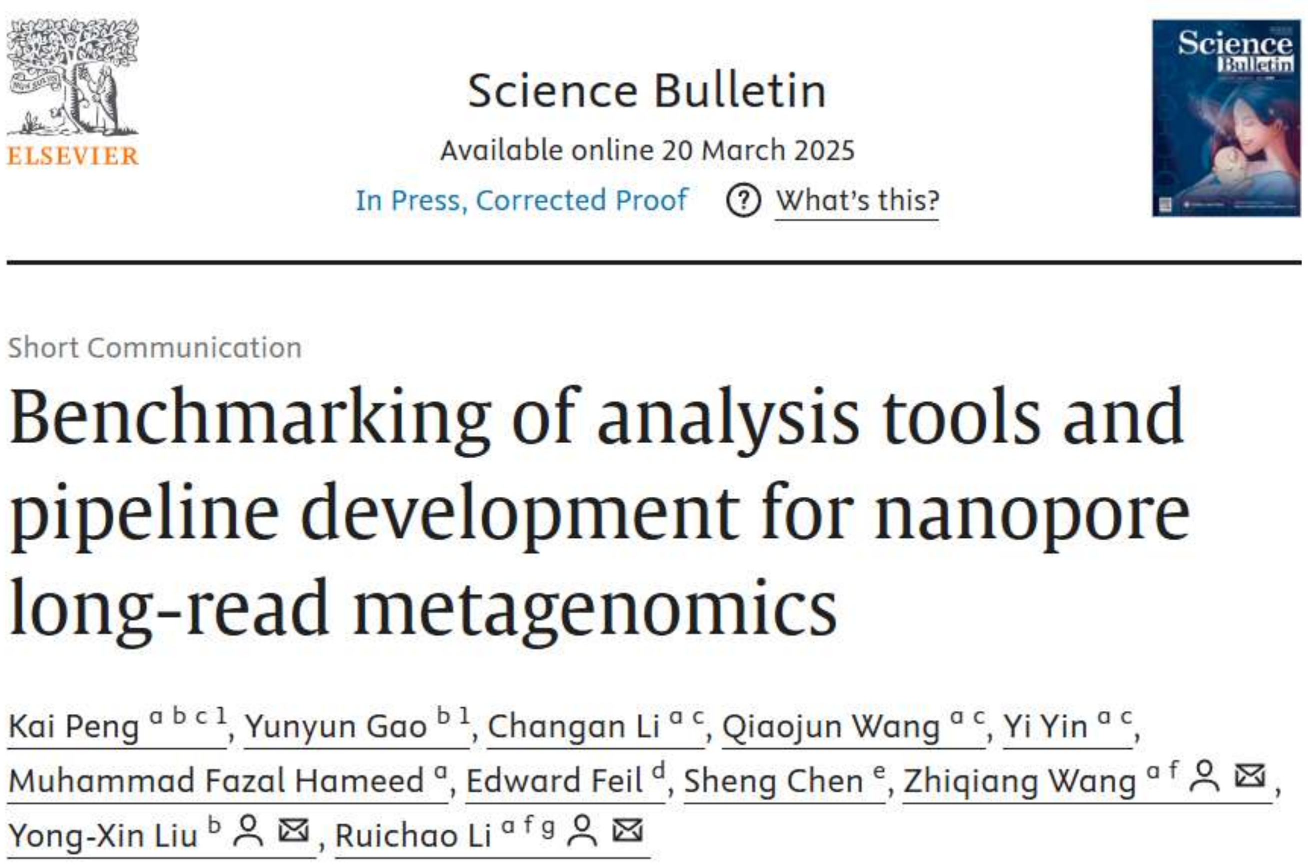


加强了不可培养微生物基因组完成图的构建；可以在reads水平进行基因功能及遗传环境研究。

Lei Liu, etc. *Microbiome*. 2022

2. EasyNanoMeta流程使用

EasyNanoMeta介绍



Science Bulletin

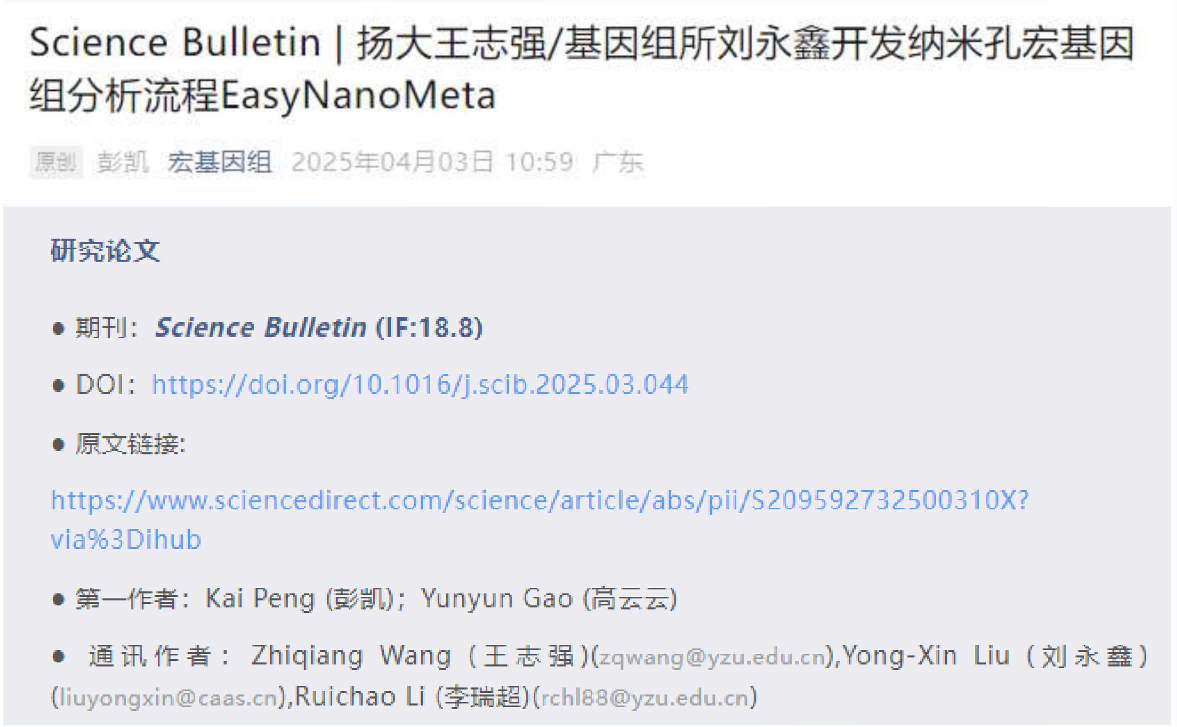
Available online 20 March 2025

In Press, Corrected Proof ⓘ What's this?

Short Communication

Benchmarking of analysis tools and pipeline development for nanopore long-read metagenomics

Kai Peng [a b c 1], Yunyun Gao [b 1], Changan Li [a c], Qiaojun Wang [a c], Yi Yin [a c], Muhammad Fazal Hameed [a], Edward Feil [d], Sheng Chen [e], Zhiqiang Wang [a f], Yong-Xin Liu [b], Ruichao Li [a f g]



Science Bulletin | 扬大王志强/基因组所刘永鑫开发纳米孔宏基因组分析流程EasyNanoMeta

原创 彭凯 宏基因组 2025年04月03日 10:59 广东

研究论文

- 期刊：***Science Bulletin*** **(IF:18.8)**
- DOI：https://doi.org/10.1016/j.scib.2025.03.044
- 原文链接:

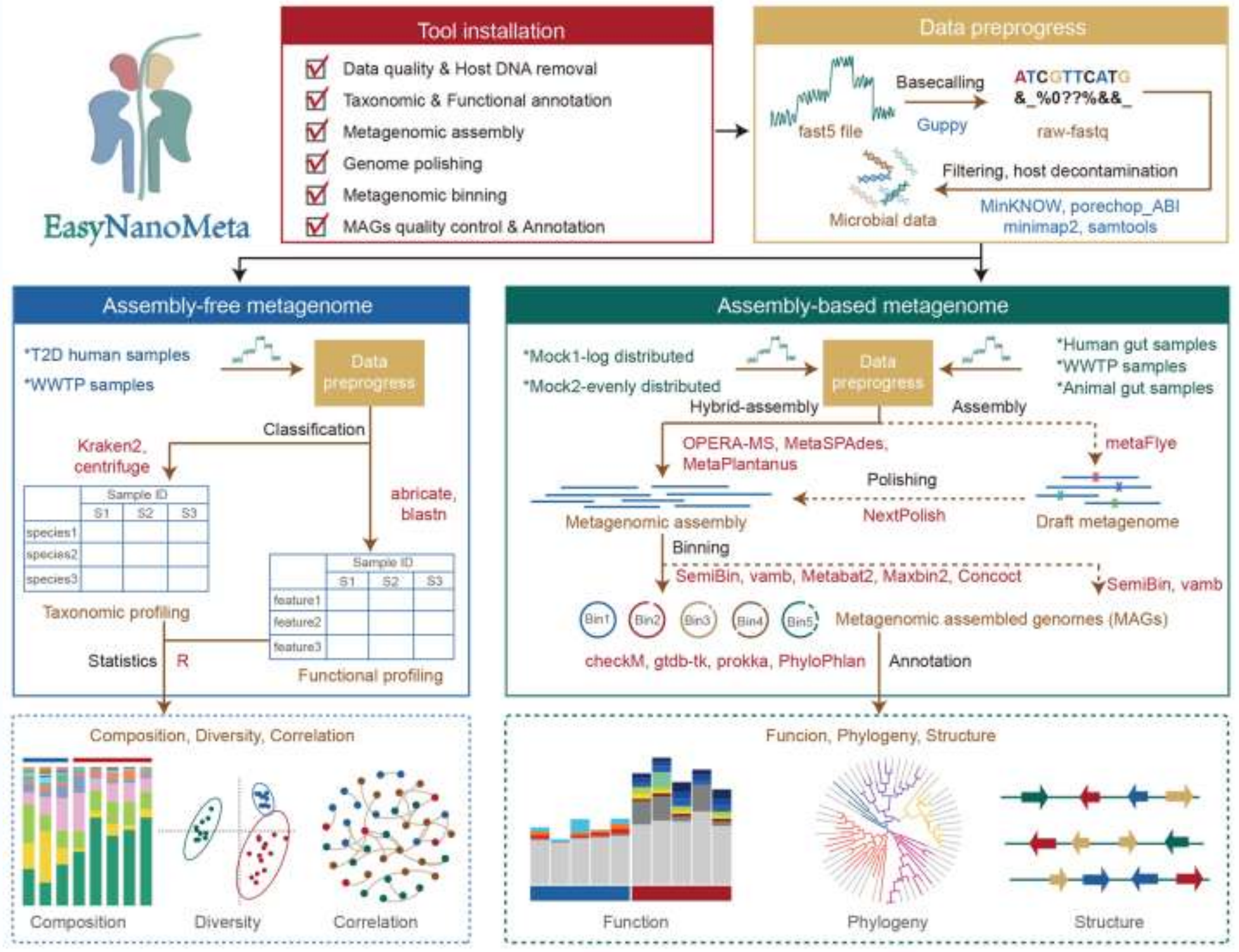
https://www.sciencedirect.com/science/article/abs/pii/S209592732500310X?via%3Dihub

- 第一作者：Kai Peng (彭凯)；Yunyun Gao (高云云)
- 通讯作者： Zhiqiang Wang（王志强)(zqwang@yzu.edu.cn),Yong-Xin Liu（刘永鑫）(liuyongxin@caas.cn),Ruichao Li (李瑞超)(rchl88@yzu.edu.cn)

Science Bulletin | 扬大王志强/基因组所刘永鑫开发纳米孔宏基因组分析流程EasyNanoMeta

Kai Peng, …, **Yong-Xin Liu, Ruichao Li.** Benchmarking of analysis tools and pipeline development for nanopore long-read metagenomics. ***Science Bulletin***. 2025. https://doi.org/10.1016/j.scib.2025.03.044

纳米孔长读宏基因组数据的分析思路



Peng, et al. ***Science Bulletin*** 2025. https://doi.org/10.1016/j.scib.2025.03.044

EasyNanoMeta：下载安装

- 下载包括所有分析软件的singularity sandbox及EasyNanoMeta
- Sandbox地址：

https://figshare.com/articles/software/A_singularity_sandbox_for_EasyNanoMeta_/27014869?file=49175110

- EasyNanoMeta软件地址：
  https://github.com/P-kai/EasyNanoMeta/archive/refs/tags/v1.0.1.tar.gz

```
cd ~/tools
wget -c https://figshare.com/ndownloader/files/49175110
wget -c https://github.com/P-kai/EasyNanoMeta/archive/refs/tags/v1.0.1.tar.gz
```

EasyNanoMeta：数据库配置

```
mkdir ~/db
cd ~/db
#human_genome
wget https://ftp.ncbi.nih.gov/genomes/all/GCA/000/001/405/GCA_000001405.28_GRCh38.p13/C
gunzip GCA_000001405.28_GRCh38.p13_genomic.fna.gz
#kraken2
wget https://genome-idx.s3.amazonaws.com/kraken/k2_standard_20230605.tar.gz
tar -zcvf k2_standard_20230605.tar.gz -C ~/db/k2_standard/
#centrifuge
wget https://zenodo.org/record/3732127/files/h%2Bp%2Bv%2Bc.tar.gz?download=1
tar -zxvf centrifuge_h+p+v.tar.gz
#checkm2
wget https://zenodo.org/record/5571251/files/checkm2_database.tar.gz
tar -zxvf checkm2_database.tar.gz
#gtdbtk
wget -c https://data.ace.uq.edu.au/public/gtdb/data/releases/latest/auxillary_files/gtdbtk_data.tar.gz
tar -zxvf gtdbtk_data.tar.gz
```

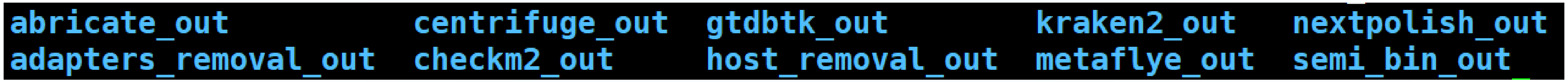
EasyNanoMeta：使用案例

使用案例：

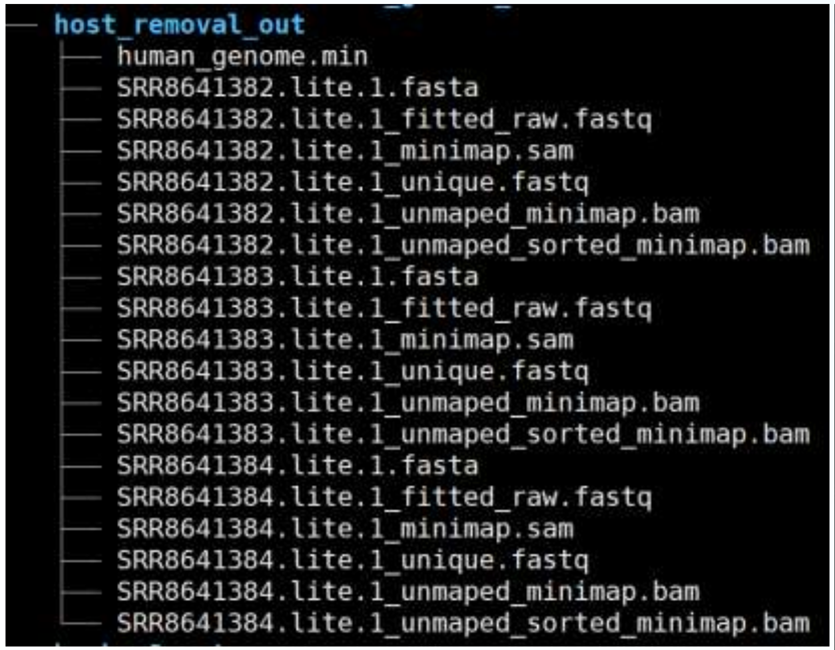
```
./easynanometa_v1/easynanometa.py -f seq/ \
-sif easynanometa.sif -t 40 \

-host-removal-reference ~/db/human_genome/human_genome/ \

-centrifuge-db ~/db/centrifuge/ \

-kraken2-db ~/db/kraken2_db/k2-standard/ \

-checkm2-db ~/db/CheckM2_database/ \

-gtdbtk-db ~/db/gtdbtk/release214/
```

EasyNanoMeta：结果展示

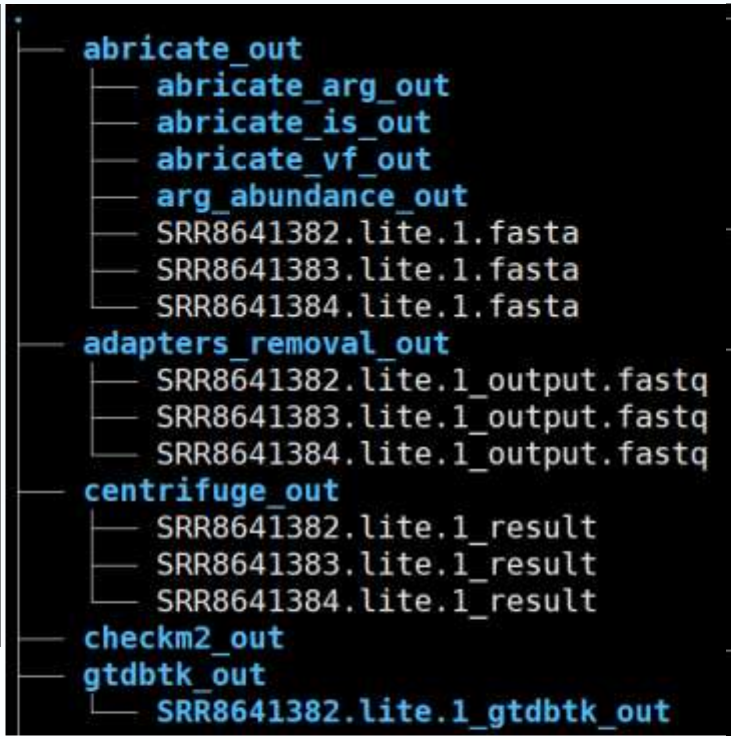
EasyNanoMeta流程输出结果：



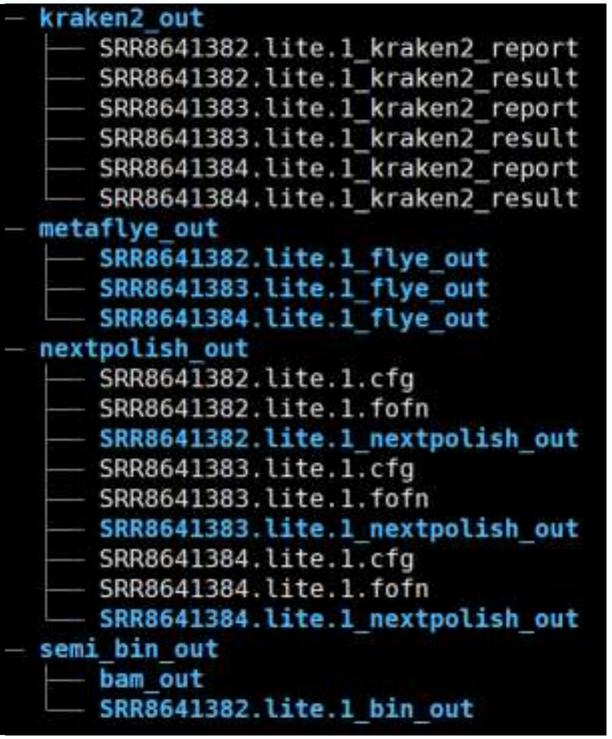
```
abricate_out            centrifuge_out  gtdbtk_out          kraken2_out     nextpolish_out
adapters_removal_out    checkm2_out     host_removal_out    metaflye_out    semi_bin_out
```



```
─ host_removal_out
  ├── human_genome.min
  ├── SRR8641382.lite.1.fasta
  ├── SRR8641382.lite.1_fitted_raw.fastq
  ├── SRR8641382.lite.1_minimap.sam
  ├── SRR8641382.lite.1_unique.fastq
  ├── SRR8641382.lite.1_unmaped_minimap.bam
  ├── SRR8641382.lite.1_unmaped_sorted_minimap.bam
  ├── SRR8641383.lite.1.fasta
  ├── SRR8641383.lite.1_fitted_raw.fastq
  ├── SRR8641383.lite.1_minimap.sam
  ├── SRR8641383.lite.1_unique.fastq
  ├── SRR8641383.lite.1_unmaped_minimap.bam
  ├── SRR8641383.lite.1_unmaped_sorted_minimap.bam
  ├── SRR8641384.lite.1.fasta
  ├── SRR8641384.lite.1_fitted_raw.fastq
  ├── SRR8641384.lite.1_minimap.sam
  ├── SRR8641384.lite.1_unique.fastq
  ├── SRR8641384.lite.1_unmaped_minimap.bam
  └── SRR8641384.lite.1_unmaped_sorted_minimap.bam
```



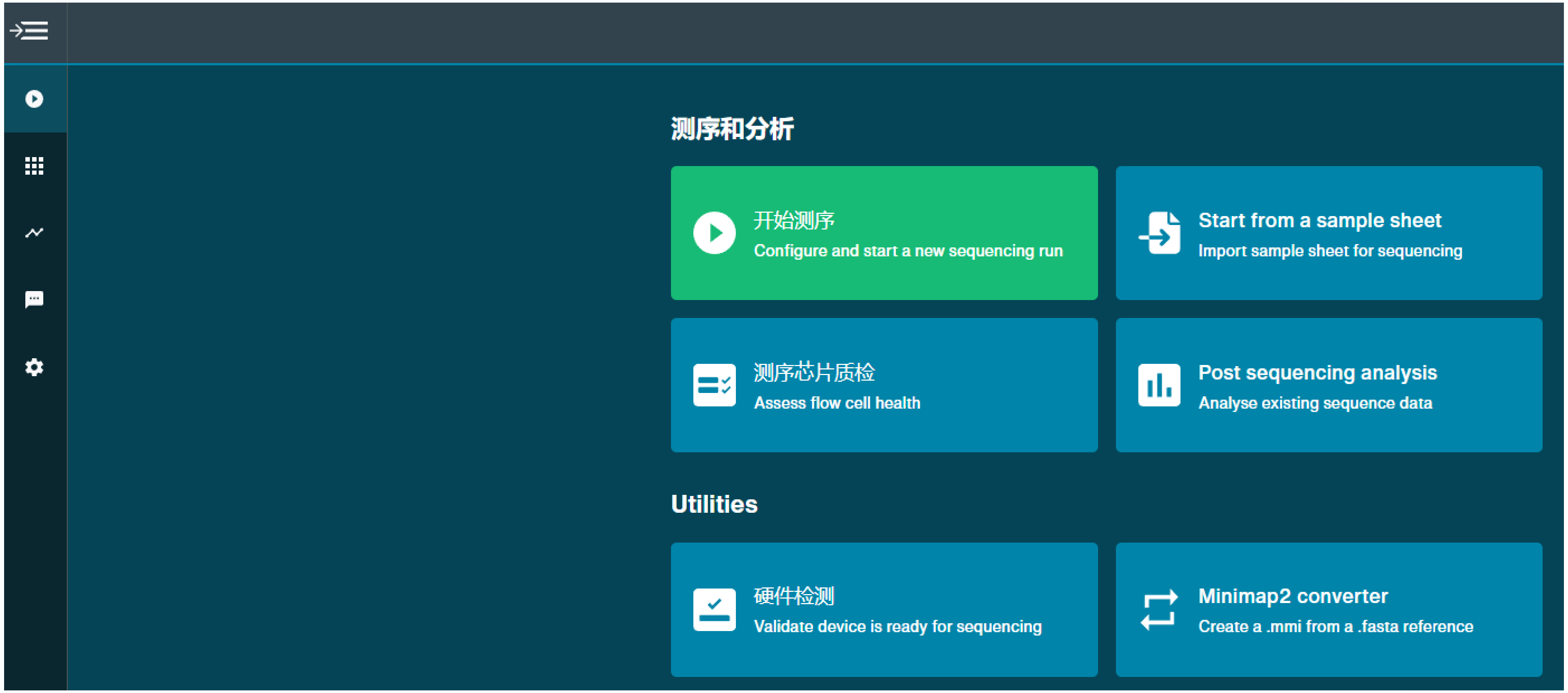
```
.
├── abricate_out
│   ├── abricate_arg_out
│   ├── abricate_is_out
│   ├── abricate_vf_out
│   ├── arg_abundance_out
│   ├── SRR8641382.lite.1.fasta
│   ├── SRR8641383.lite.1.fasta
│   └── SRR8641384.lite.1.fasta
├── adapters_removal_out
│   ├── SRR8641382.lite.1_output.fastq
│   ├── SRR8641383.lite.1_output.fastq
│   └── SRR8641384.lite.1_output.fastq
├── centrifuge_out
│   ├── SRR8641382.lite.1_result
│   ├── SRR8641383.lite.1_result
│   └── SRR8641384.lite.1_result
├── checkm2_out
├── gtdbtk_out
│   └── SRR8641382.lite.1_gtdbtk_out
```



```
─ kraken2_out
  ├── SRR8641382.lite.1_kraken2_report
  ├── SRR8641382.lite.1_kraken2_result
  ├── SRR8641383.lite.1_kraken2_report
  ├── SRR8641383.lite.1_kraken2_result
  ├── SRR8641384.lite.1_kraken2_report
  └── SRR8641384.lite.1_kraken2_result
─ metaflye_out
  ├── SRR8641382.lite.1_flye_out
  ├── SRR8641383.lite.1_flye_out
  └── SRR8641384.lite.1_flye_out
─ nextpolish_out
  ├── SRR8641382.lite.1.cfg
  ├── SRR8641382.lite.1.fofn
  ├── SRR8641382.lite.1_nextpolish_out
  ├── SRR8641383.lite.1.cfg
  ├── SRR8641383.lite.1.fofn
  ├── SRR8641383.lite.1_nextpolish_out
  ├── SRR8641384.lite.1.cfg
  ├── SRR8641384.lite.1.fofn
  └── SRR8641384.lite.1_nextpolish_out
─ semi_bin_out
  ├── bam_out
  └── SRR8641382.lite.1_bin_out
```

3. 数据质量控制及去宿主

MinKNOW进行测序管理


测序和分析

开始测序
Configure and start a new sequencing run

Start from a sample sheet
Import sample sheet for sequencing

测序芯片质检
Assess flow cell health

Post sequencing analysis
Analyse existing sequence data

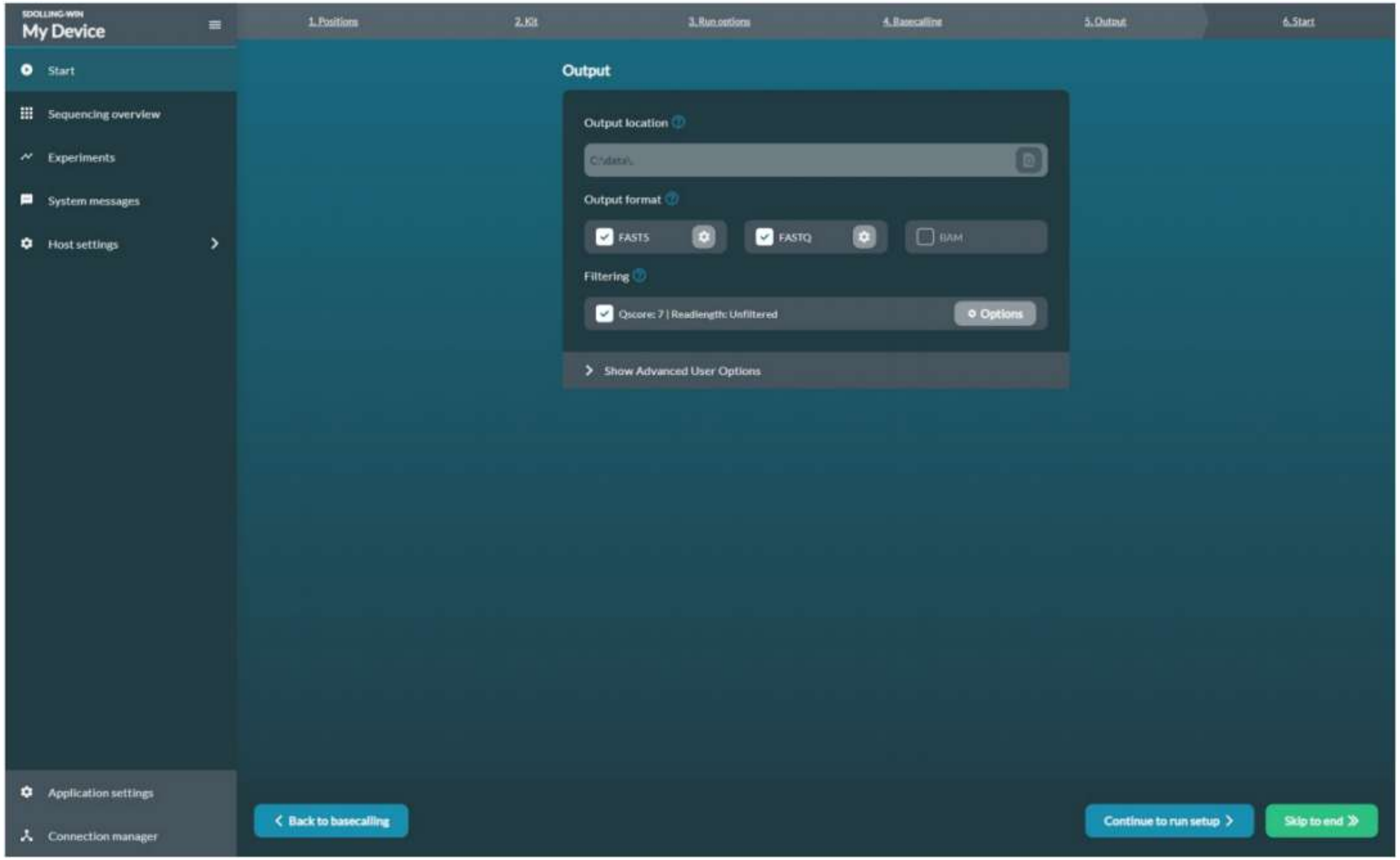
Utilities

硬件检测
Validate device is ready for sequencing

Minimap2 converter
Create a .mmi from a .fasta reference

跨平台的ONT纳米孔测序管理软件（华大、齐碳等均有自主开发的相应软件）。 https://nanoporetech.com/document/experiment-companion-minknow

MinKNOW进行数据过滤



Output

Output location

C:\data\

Output format

[✓ FAST5] [✓ FASTQ] BAM

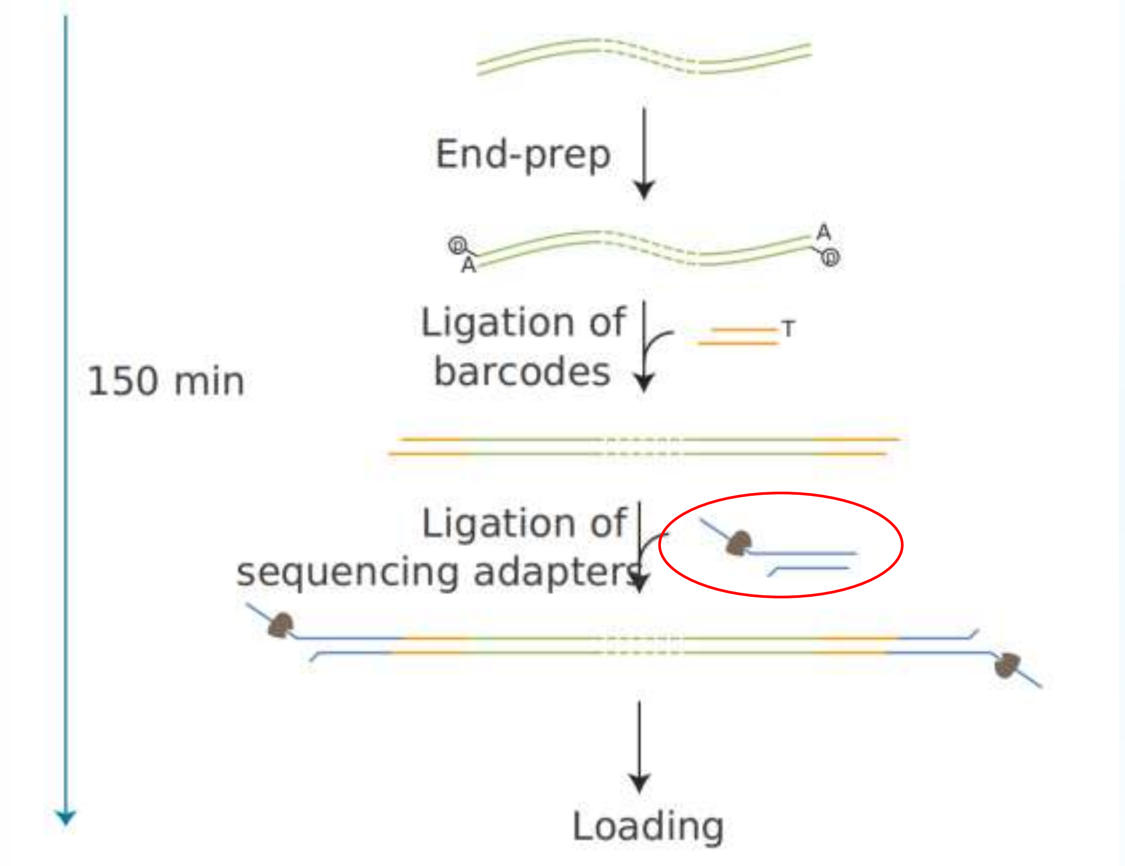
Filtering

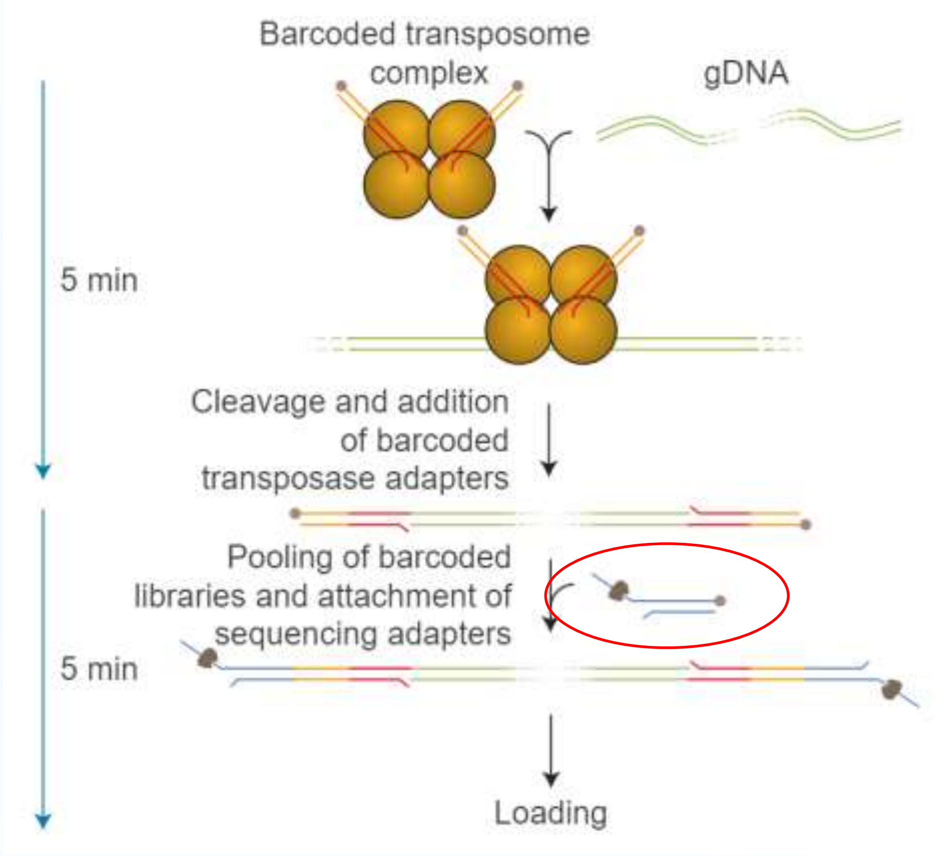
[✓ Qscore: 7 | Readlength: Unfiltered]

Filtering 选 项 :

根据测序数据的读长和Q值等对数据进行基础过滤。

porechop_abi去接头




Barcoded transposome complex
gDNA
5 min
Cleavage and addition of barcoded transposase adapters
Pooling of barcoded libraries and attachment of sequencing adapters
5 min
Loading

连接法建库

转座酶法建库

porechop_abi：去除纳米孔测序数据接头

- 安装

```
conda create -y -n porechop_abi
conda activate porechop_abi
conda install -f -c conda-forge -c bioconda  porechop_abi
```

- 示例：使用porechop_abi进行nanopore数据接头去除
- 激活软件所在环境

```
conda activate porechop_abi
```

- 设置输入样本名称

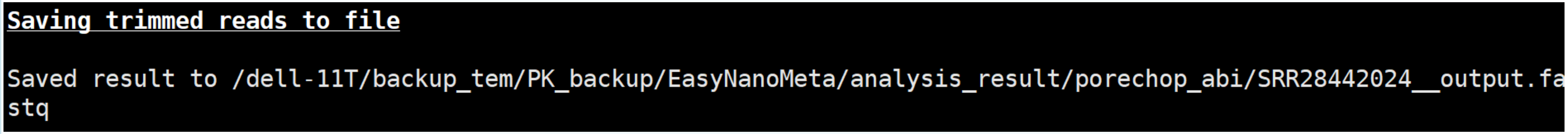
```
i=sample_name
```

- 使用默认参数去除接头

```
porechop_abi --ab_initio -i ${i}.fastq  -o ${i}_output.fastq  -t 24
```

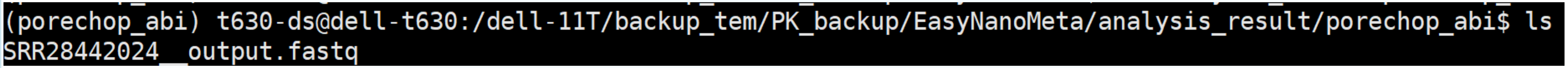
porechop_abi：结果展示

运行结束标识



```
Saving trimmed reads to file

Saved result to /dell-11T/backup_tem/PK_backup/EasyNanoMeta/analysis_result/porechop_abi/SRR28442024__output.fa
stq
```

结果文件



```
(porechop_abi) t630-ds@dell-t630:/dell-11T/backup_tem/PK_backup/EasyNanoMeta/analysis_result/porechop_abi$ ls
SRR28442024__output.fastq
```

porechop_abi：多样本数据去接头

- 生成所有样本的名称

```
ls *1.fastq && cut -f1 -d '.' > samples_name
```

- 生成批量运行的脚本

```
for i in `cat samples_name`; do echo "porechop_abi --ab_initio -1 ${i}.fastq -o ${i}_output.fastq -t 24";  done > porechop_abi.sh
```

- 运行批量执行脚本

```
sh porechop_abi.sh
```

NanoPack: 统计纳米孔测序数据基础信息

- 安装

```
conda create -y -n nanopack python=3.10
conda activate nanopack
```

- 在conda的nanopack环境中，使用pip安装nanopack

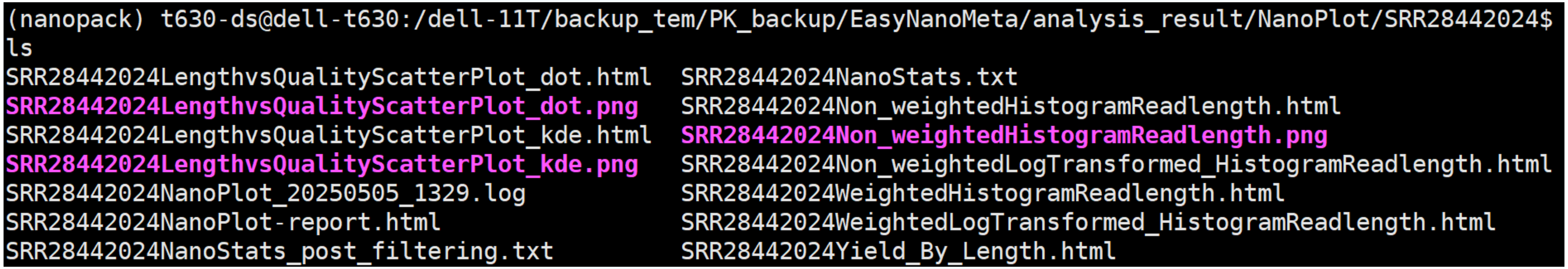
```
pip install nanopack
```

- 示例:

```
i=sample_name
NanoPlot --fastq ${i}.fastq.gz -t 12 -p ${i} --color blue --plots hex dot kde -o
nanoplot

i1=sample_name1
i2=sample_name2
NanoComp --dpi 500 -t 24 -p prefix -f svg --fastq ${i1}.fastq ${i2}.fastq --
names ${i1} ${i2} -o NanoComp
```

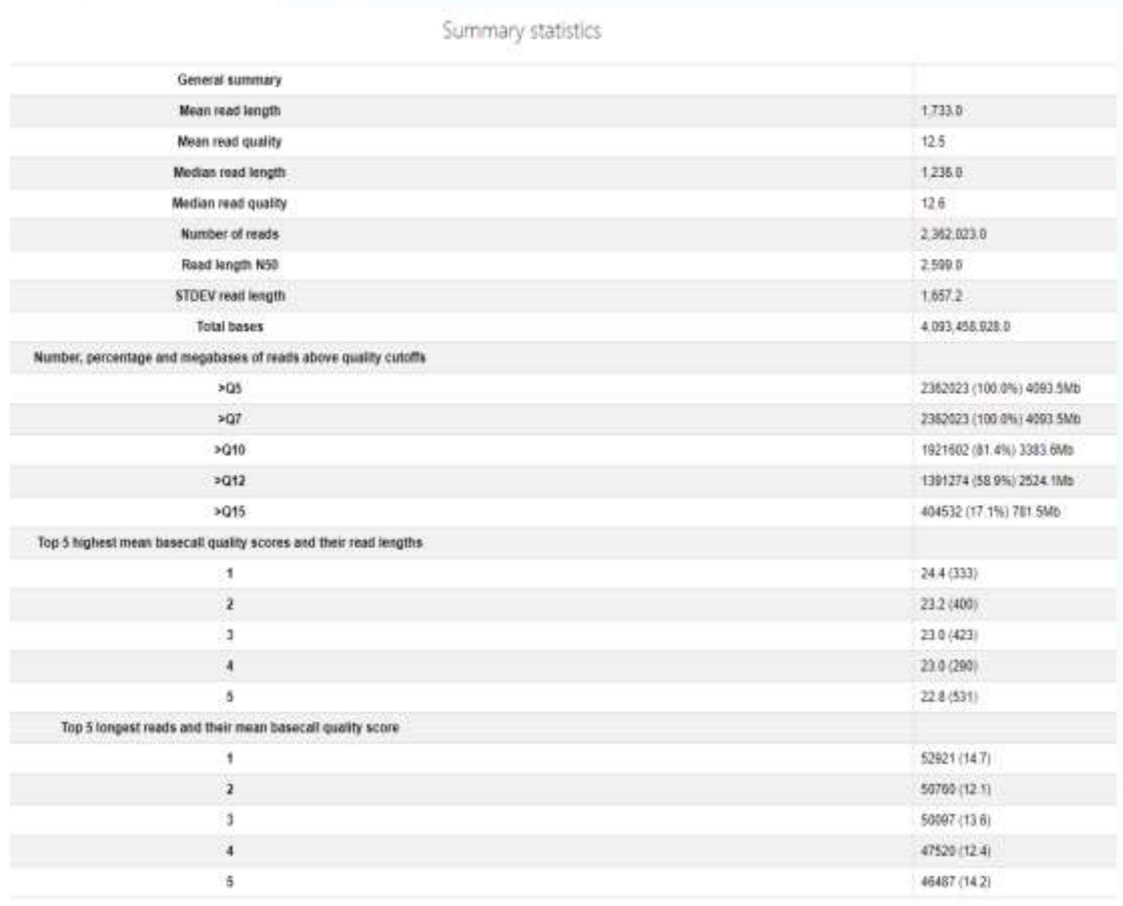
NanoPack: 结果目录

NanoPlot输出目录文件



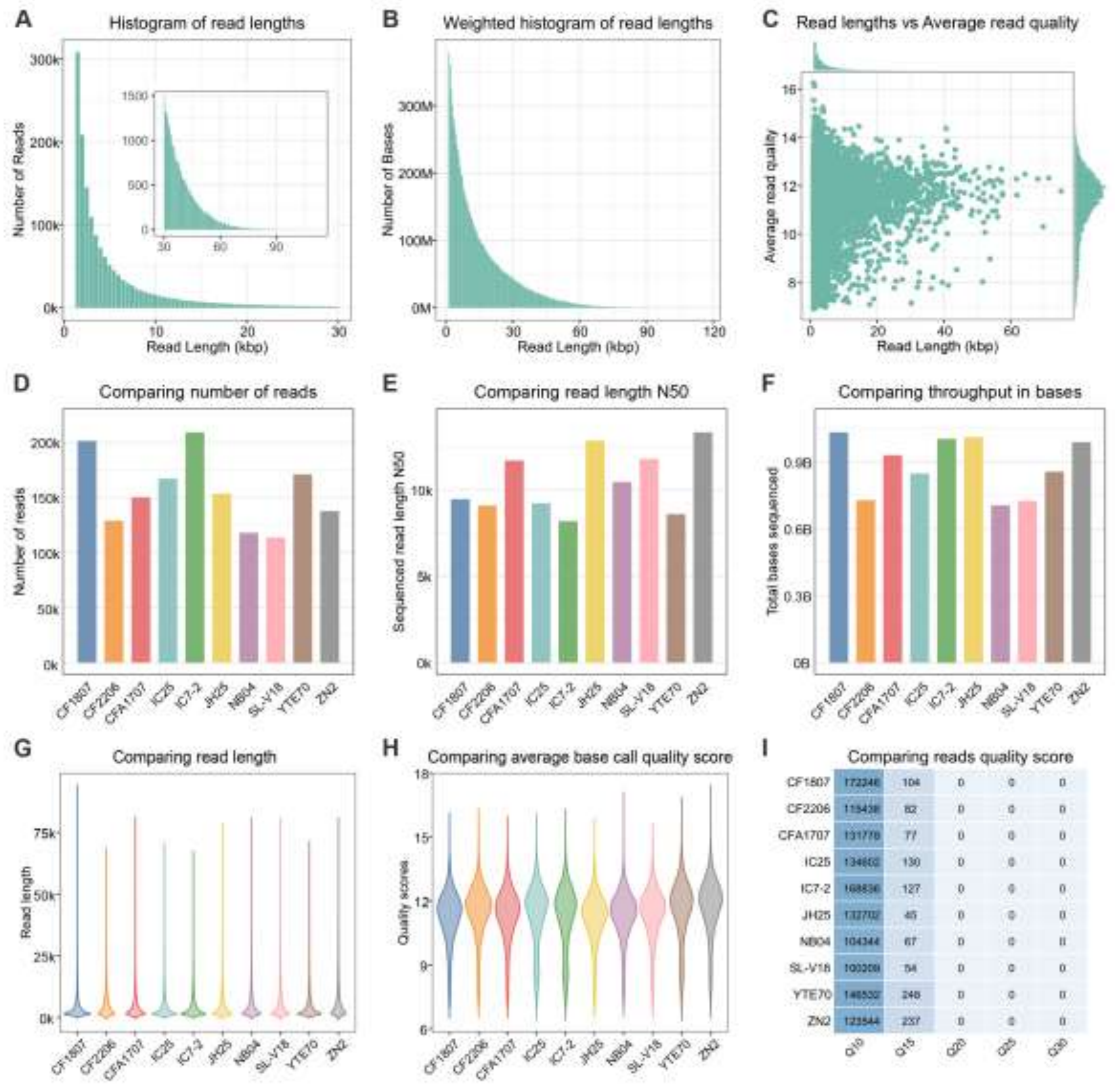
```
(nanopack) t630-ds@dell-t630:/dell-11T/backup_tem/PK_backup/EasyNanoMeta/analysis_result/NanoPlot/SRR28442024$
ls
SRR28442024LengthvsQualityScatterPlot_dot.html   SRR28442024NanoStats.txt
SRR28442024LengthvsQualityScatterPlot_dot.png    SRR28442024Non_weightedHistogramReadlength.html
SRR28442024LengthvsQualityScatterPlot_kde.html   SRR28442024Non_weightedHistogramReadlength.png
SRR28442024LengthvsQualityScatterPlot_kde.png    SRR28442024Non_weightedLogTransformed_HistogramReadlength.html
SRR28442024NanoPlot_20250505_1329.log            SRR28442024WeightedHistogramReadlength.html
SRR28442024NanoPlot-report.html                  SRR28442024WeightedLogTransformed_HistogramReadlength.html
SRR28442024NanoStats_post_filtering.txt          SRR28442024Yield_By_Length.html
```

NanoPack: 结果文件展示



Summary statistics

General summary	
Mean read length	1,733.0
Mean read quality	12.5
Median read length	1,236.0
Median read quality	12.6
Number of reads	2,362,023.0
Read length N50	2,599.0
STDEV read length	1,657.2
Total bases	4,093,458,928.0
Number, percentage and megabases of reads above quality cutoffs	
>Q5	2362023 (100.0%) 4093.5Mb
>Q7	2362023 (100.0%) 4093.5Mb
>Q10	1921602 (81.4%) 3383.6Mb
>Q12	1391274 (58.9%) 2524.1Mb
>Q15	404532 (17.1%) 781.5Mb
Top 5 highest mean basecall quality scores and their read lengths	
1	24.4 (333)
2	23.2 (400)
3	23.0 (423)
4	23.0 (290)
5	22.8 (531)
Top 5 longest reads and their mean basecall quality score	
1	52921 (14.7)
2	50760 (12.1)
3	50097 (13.6)
4	47520 (12.4)
5	46487 (14.2)



minimap2, samtools, bedtools：三代数据去宿主

- 软件安装

```
conda create -y -n host_removal
conda activate host_removal
conda install -c bioconda minimap2
conda install -c bioconda samtools
conda install -c bioconda bedtools
```

- 使用conda package进行软件安装
- conda package下
  https://figshare.com/articles/software/host_removal/25569159?file=45553653

```
mkdir ~/miniconda3/envs/host_removal/
tar -xzvf host_removal.tar.gz -C ~/miniconda3/envs/host_removal/
conda activate host_removal
conda unpack
```

minimap2, samtools, bedtools：数据库配置

- 数据库配置

```
db=~/db
mkdir -p ${db} && cd ${db}
```

- 人类基因组下载

```
wget
https://ftp.ncbi.nih.gov/genomes/all/GCA/000/001/405/GCA_000001405.28_GRC
h38.p13/C

gunzip GCA_000001405.28_GRCh38.p13_genomic.fna.gz
```

minimap2, samtools, bedtools：软件使用

- 进入去宿主的conda环境

```
conda activate host_removal
```

- 建立minimap2比对索引，human基因组为参考对齐序列

```
i=human_genome
minimap2 -d ${i}.min ${i}.fasta
```

- 使用minimap2进行数据比对

```
minimap2 -ax map-ont -t 24 ${i}.min ../raw.fasta -o minimap.sam
```

- 提取未匹配到宿主的序列

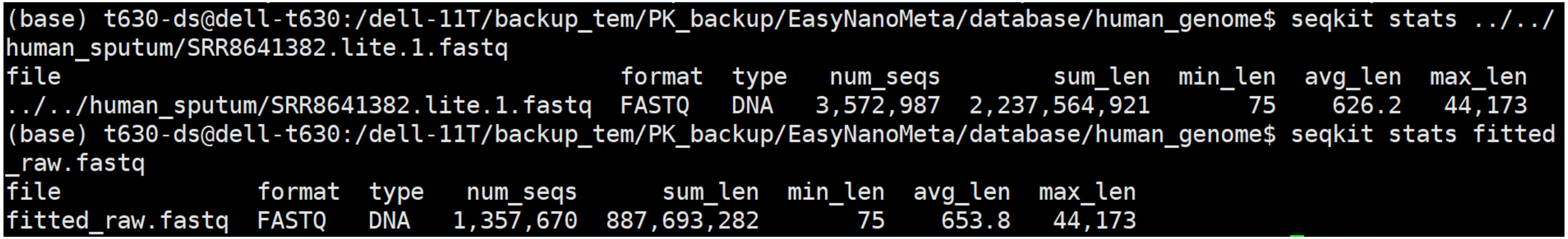
```
samtools view -bS -T -@24 ${i}.fasta -f 4 minimap.sam > unmaped_minimap.bam
```

- 将bam文件转换为fastq文件，首先对bam文件进行排序，然后使用bedtools中的bamtofastq进行bam文件到fastq文件的转换

```
samtools sort -n unmaped_minimap.bam -o unmaped_sorted_minimap.bam
bedtools bamtofastq -i unmaped_sorted_minimap.bam -fq fitted_raw.fastq
```

minimap2, samtools, bedtools：结果展示

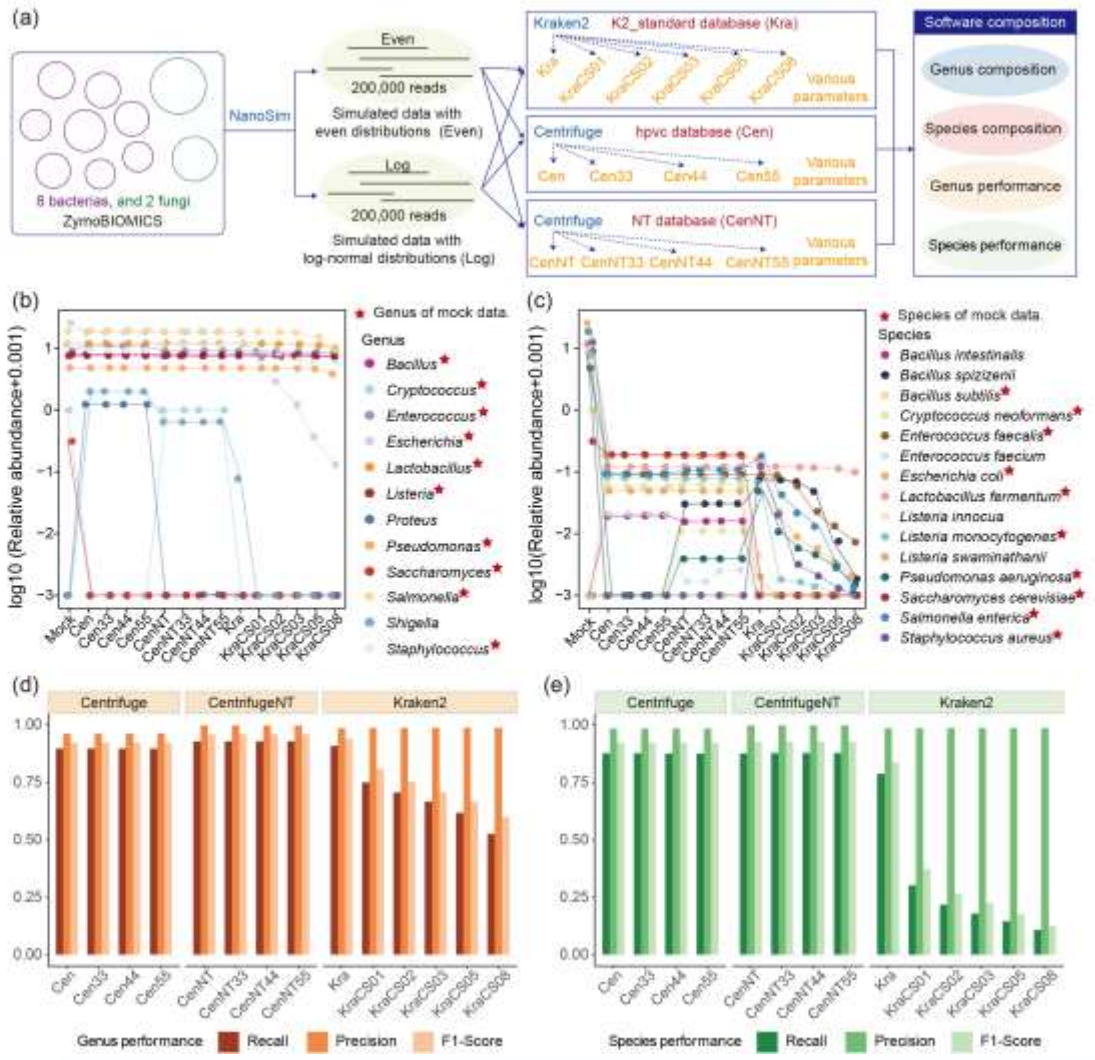
去宿主前后数据信息比对



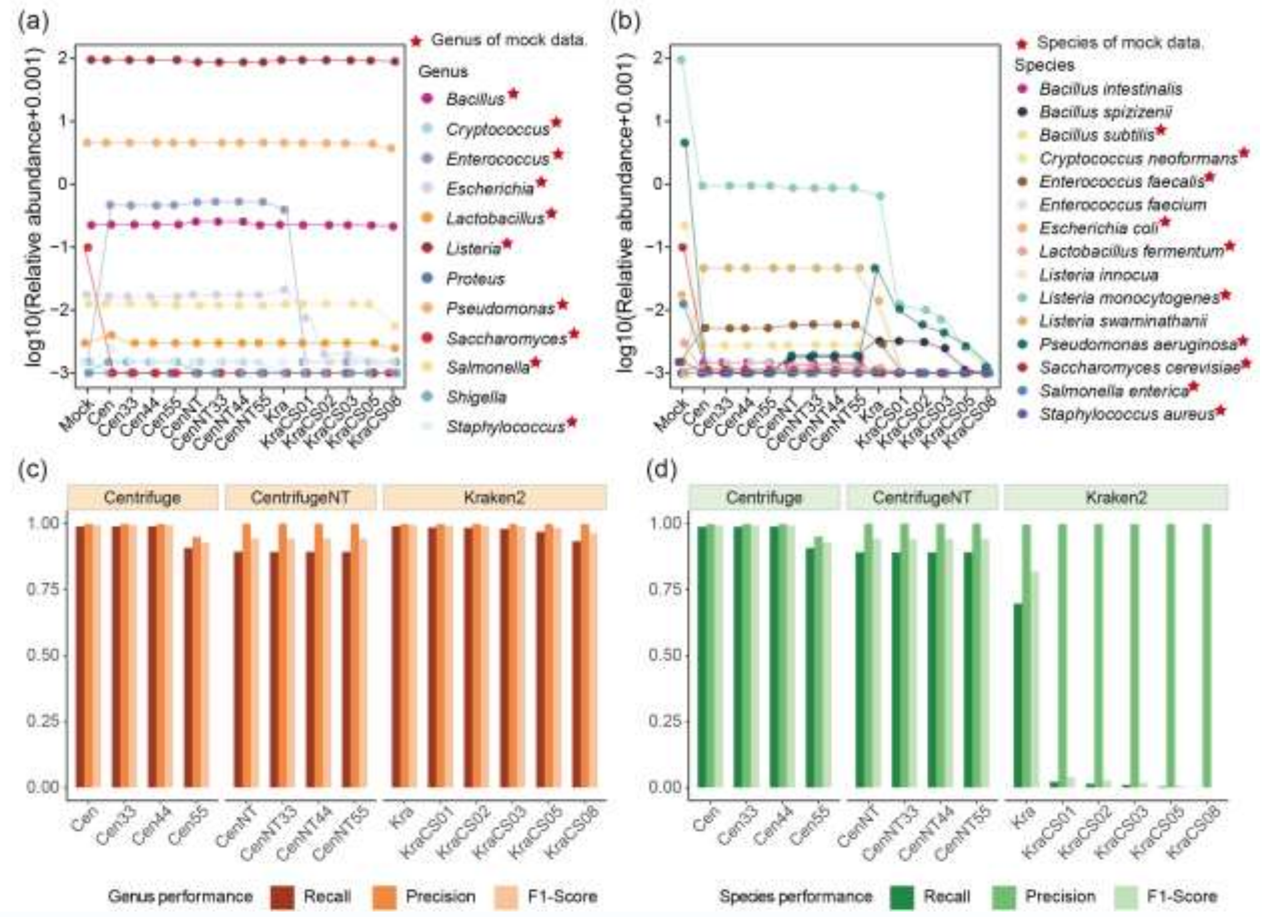
```
(base) t630-ds@dell-t630:/dell-11T/backup_tem/PK_backup/EasyNanoMeta/database/human_genome$ seqkit stats ../../
human_sputum/SRR8641382.lite.1.fastq
file                                         format  type    num_seqs        sum_len  min_len  avg_len  max_len
../../human_sputum/SRR8641382.lite.1.fastq  FASTQ   DNA   3,572,987  2,237,564,921       75    626.2   44,173
(base) t630-ds@dell-t630:/dell-11T/backup_tem/PK_backup/EasyNanoMeta/database/human_genome$ seqkit stats fitted
_raw.fastq
file                 format  type    num_seqs      sum_len  min_len  avg_len  max_len
fitted_raw.fastq    FASTQ   DNA   1,357,670  887,693,282       75    653.8   44,173
```

4. 物种注释及功能注释

软件如何选择?



(a)
8 bacterias, and 2 fungi
ZymoBIOMICS
NanoSim
Even
200,000 reads
Simulated data with even distributions (Even)
Log
200,000 reads
Simulated data with log-normal distributions (Log)
Kraken2 K2_standard database (Kra)
Kra KraCS01 KraCS02 KraCS03 KraCS05 KraCS08 Various parameters
Centrifuge hpvc database (Cen)
Cen Cen33 Cen44 Cen55 Various parameters
Centrifuge NT database (CenNT)
CenNT CenNT33 CenNT44 CenNT55 Various parameters
Software composition
Genus composition
Species composition
Genus performance
Species performance
(b) (c) (d) (e)
log10 (Relative abundance+0.001)
Genus of mock data
Species of mock data
Genus performance Recall Precision F1-Score
Species performance Recall Precision F1-Score
Centrifuge CentrifugeNT Kraken2



(a) (b) (c) (d)
log10(Relative abundance+0.001)
Genus of mock data
Genus
Bacillus
Cryptococcus
Enterococcus
Escherichia
Lactobacillus
Listeria
Proteus
Pseudomonas
Saccharomyces
Salmonella
Shigella
Staphylococcus
Species of mock data
Species
Bacillus intestinalis
Bacillus spizizenii
Bacillus subtilis
Cryptococcus neoformans
Enterococcus faecalis
Enterococcus faecium
Escherichia coli
Lactobacillus fermentum
Listeria innocua
Listeria monocytogenes
Listeria swaminathanii
Pseudomonas aeruginosa
Saccharomyces cerevisiae
Salmonella enterica
Staphylococcus aureus
Centrifuge CentrifugeNT Kraken2
Genus performance Recall Precision F1-Score
Species performance Recall Precision F1-Score

centrifuge：纳米孔长读数据物种注释

- 软件安装

```
cd ~/tools
wget https://github.com/DaehwanKimLab/centrifuge/archive/refs/tags/v1.0.4.tar.gz
tar -zxvf v1.0.4.tar.gz
cd centrifuge-1.0.4
make
make install prefix=~/tools/centrifuge-1.0.4
```

- Github克隆安装

```
git clone https://github.com/DaehwanKimLab/centrifuge
cd centrifuge
make
make install prefix=~/tools/centrifuge-1.0.4
```

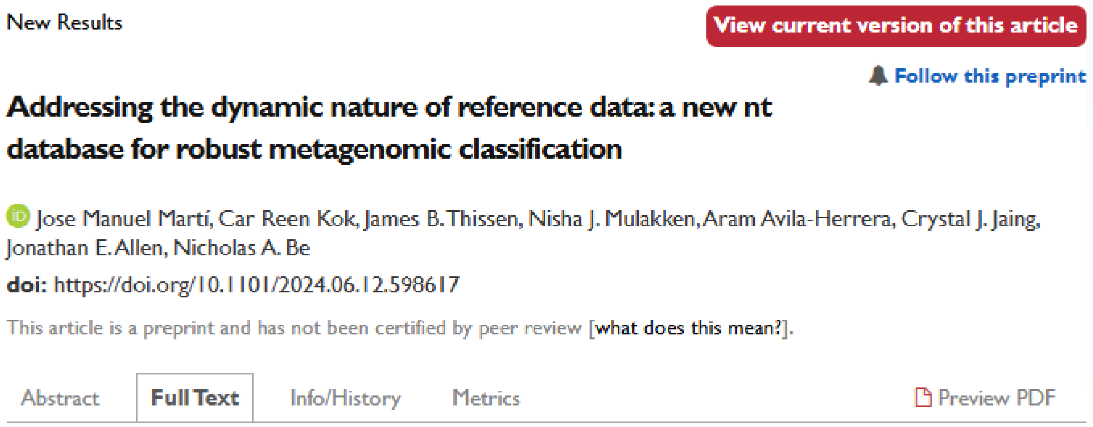
centrifuge：配置数据库

- 配置软件自有数据库

```
cd ~/db
wget https://zenodo.org/record/3732127/files/h%2Bp%2Bv%2Bc.tar.gz?download=1
tar -zxvf centrifuge_h+p+v.tar.gz
```

- 配置nt数据库（可通过filezilla进行下载）

ftp://gdo-bioinformatics.ucllnl.org/centrifuge/nt_wntr23



New Results

Addressing the dynamic nature of reference data: a new nt database for robust metagenomic classification

Jose Manuel Martí, Car Reen Kok, James B. Thissen, Nisha J. Mulakken, Aram Avila-Herrera, Crystal J. Jaing, Jonathan E. Allen, Nicholas A. Be

doi: https://doi.org/10.1101/2024.06.12.598617

This article is a preprint and has not been certified by peer review [what does this mean?].

centrifuge：使用方法

- 使用centrifuge进行微生物物种组成分析，输入文件为fastq数据

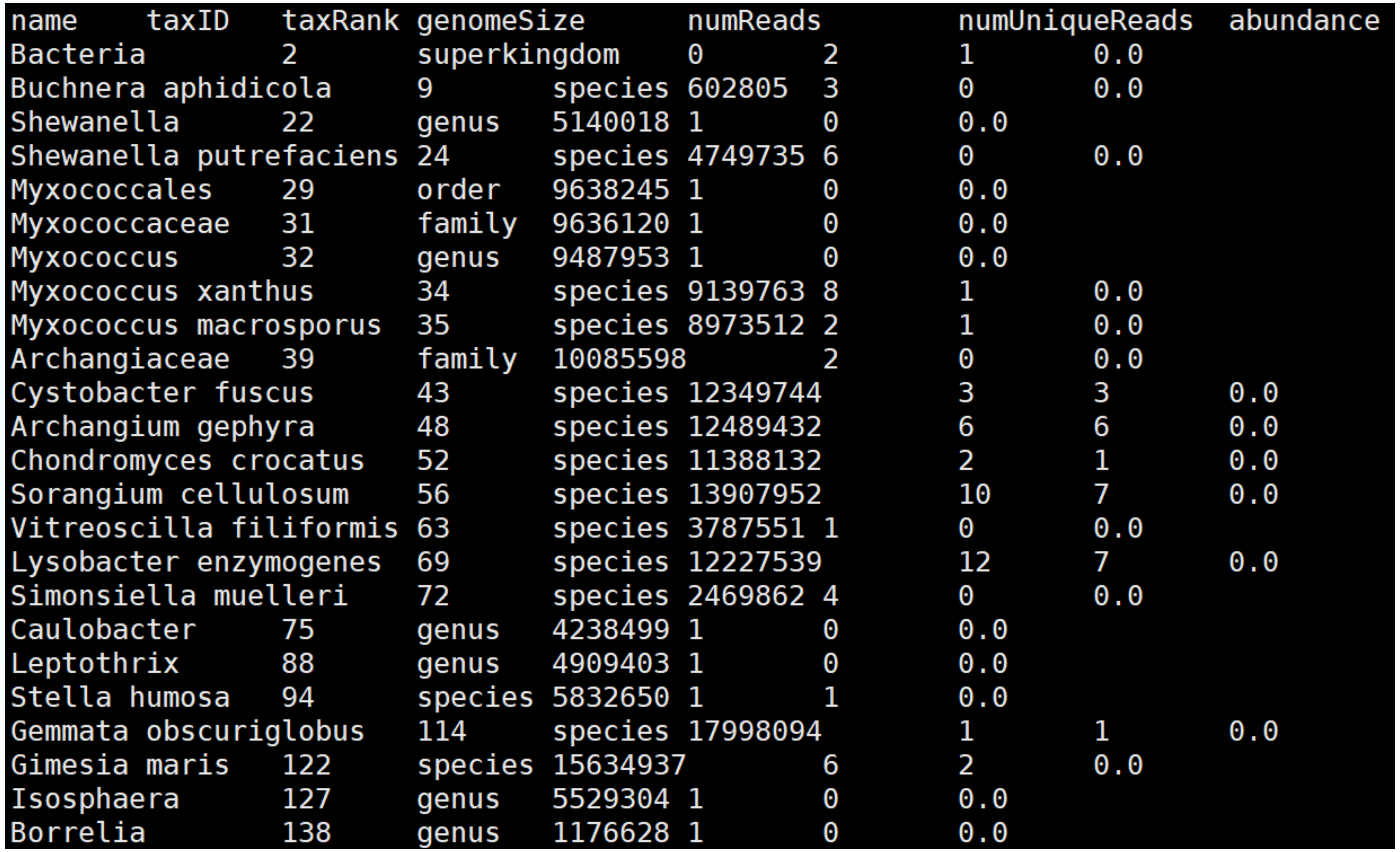
```
i=sample_name
~/Pengkai/Tools/centrifuge/bin/centrifuge -p 24 -x
/path/to/Database/centrifuge_h+p+v_20200318/hpv -q ${i}.fastq   --report-file
${i}_report   -S ${i}_result
```

- 使用centrifuge进行微生物物种组成分析，输入文件为fasta数据

```
i=sample_name
~/Pengkai/Tools/centrifuge/bin/centrifuge -p 24  -x
/path/to/Database/centrifuge_h+p+v_20200318/hpv  -f ${i}.fasta  --report-file
${i}_report  -S ${i}_result
```

centrifuge：结果文件

1、比对上物种名字，如果鉴定不到种，则上升一级；
2、物种分类 ID；
3、物种分类层级 rank；
4、对应基因组大小；
5、比对到的 reads 数目，包括多重比对的结果；
6、唯一比对上的 reads 数目；
7、比对的丰度，比对上区域/基因组长度。



```
name    taxID   taxRank genomeSize      numReads        numUniqueReads  abundance
Bacteria        2       superkingdom    0       2       1       0.0
Buchnera aphidicola     9       species 602805  3       0       0.0
Shewanella      22      genus   5140018 1       0       0.0
Shewanella putrefaciens 24      species 4749735 6       0       0.0
Myxococcales    29      order   9638245 1       0       0.0
Myxococcaceae   31      family  9636120 1       0       0.0
Myxococcus      32      genus   9487953 1       0       0.0
Myxococcus xanthus      34      species 9139763 8       1       0.0
Myxococcus macrosporus  35      species 8973512 2       1       0.0
Archangiaceae   39      family  10085598        2       0       0.0
Cystobacter fuscus      43      species 12349744        3       3       0.0
Archangium gephyra      48      species 12489432        6       6       0.0
Chondromyces crocatus   52      species 11388132        2       1       0.0
Sorangium cellulosum    56      species 13907952        10      7       0.0
Vitreoscilla filiformis 63      species 3787551 1       0       0.0
Lysobacter enzymogenes  69      species 12227539        12      7       0.0
Simonsiella muelleri    72      species 2469862 4       0       0.0
Caulobacter     75      genus   4238499 1       0       0.0
Leptothrix      88      genus   4909403 1       0       0.0
Stella humosa   94      species 5832650 1       1       0.0
Gemmata obscuriglobus   114     species 17998094        1       1       0.0
Gimesia maris   122     species 15634937        6       2       0.0
Isosphaera      127     genus   5529304 1       0       0.0
Borrelia        138     genus   1176628 1       0       0.0
```

kraken2：物种功能注释

- 软件安装（直接下载安装）

```
cd ~/tools
wget https://github.com/DerrickWood/kraken2/archive/refs/tags/v2.1.3.tar.gz
tar -zxvf v2.1.3.tar.gz
cd kraken2-2.1.3/
sh install_kraken2.sh ~/tools/kraken2-2.1.3/
```

- 软件安装（使用conda进行安装）

```
conda create -n kraken2
conda activate kraken2
conda install -y kraken2
```

kraken2：附属工具包安装

- KrakenTools软件安装（直接下载安装）

```
cd ~/tools
wget https://github.com/jenniferlu717/KrakenTools/archive/refs/tags/v1.2.tar.gz
tar -zxvf v1.2.tar.gz
```

kraken2：数据库下载配置

- 直接使用kraken2自带脚本进行数据库下载

```
kraken2-build --standard --threads 24 --db ~/db/kraken2_db
```

- 下载构建的数据库直接使用，推荐网站：https://benlangmead.github.io/aws-indexes/k2

```
cd ~/db
wget https://genome-idx.s3.amazonaws.com/kraken/k2_standard_20230605.tar.gz
```

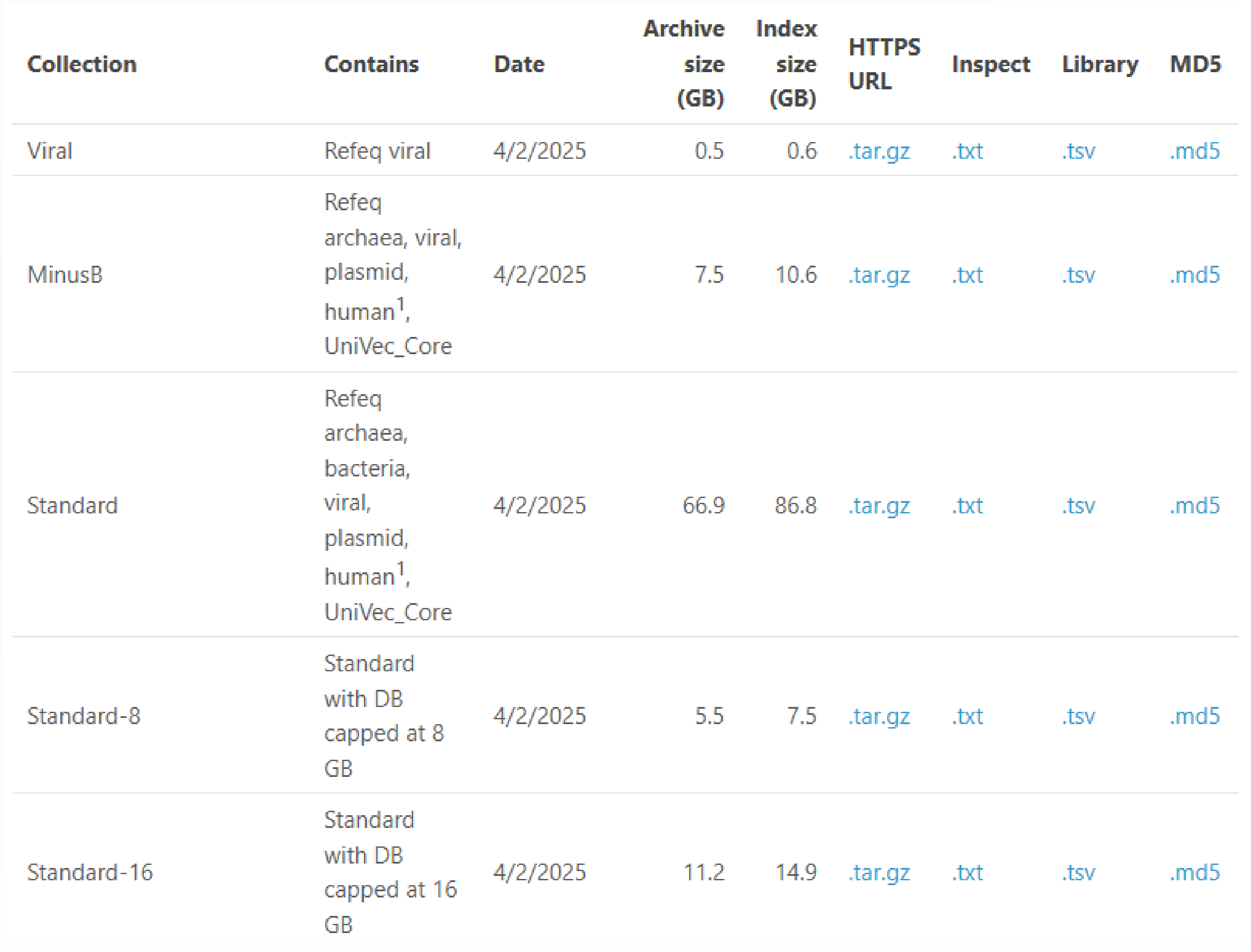
- 解压数据库到指定数据库位置

```
tar -zcvf k2_standard_20230605.tar.gz -C ~/db/k2_standard/
```

kraken2：数据库下载配置

https://benlangmead.github.io/aws-indexes/k2

网站中各类kraken2数据库，可直接下载使用



Collection	Contains	Date	Archive size (GB)	Index size (GB)	HTTPS URL	Inspect	Library	MD5
Viral	Refeq viral	4/2/2025	0.5	0.6	.tar.gz	.txt	.tsv	.md5
MinusB	Refeq archaea, viral, plasmid, human[1], UniVec_Core	4/2/2025	7.5	10.6	.tar.gz	.txt	.tsv	.md5
Standard	Refeq archaea, bacteria, viral, plasmid, human[1], UniVec_Core	4/2/2025	66.9	86.8	.tar.gz	.txt	.tsv	.md5
Standard-8	Standard with DB capped at 8 GB	4/2/2025	5.5	7.5	.tar.gz	.txt	.tsv	.md5
Standard-16	Standard with DB capped at 16 GB	4/2/2025	11.2	14.9	.tar.gz	.txt	.tsv	.md5

kraken2：使用案例

- 使用kraken2进行微生物物种组成分析，输入文件为fastq数据

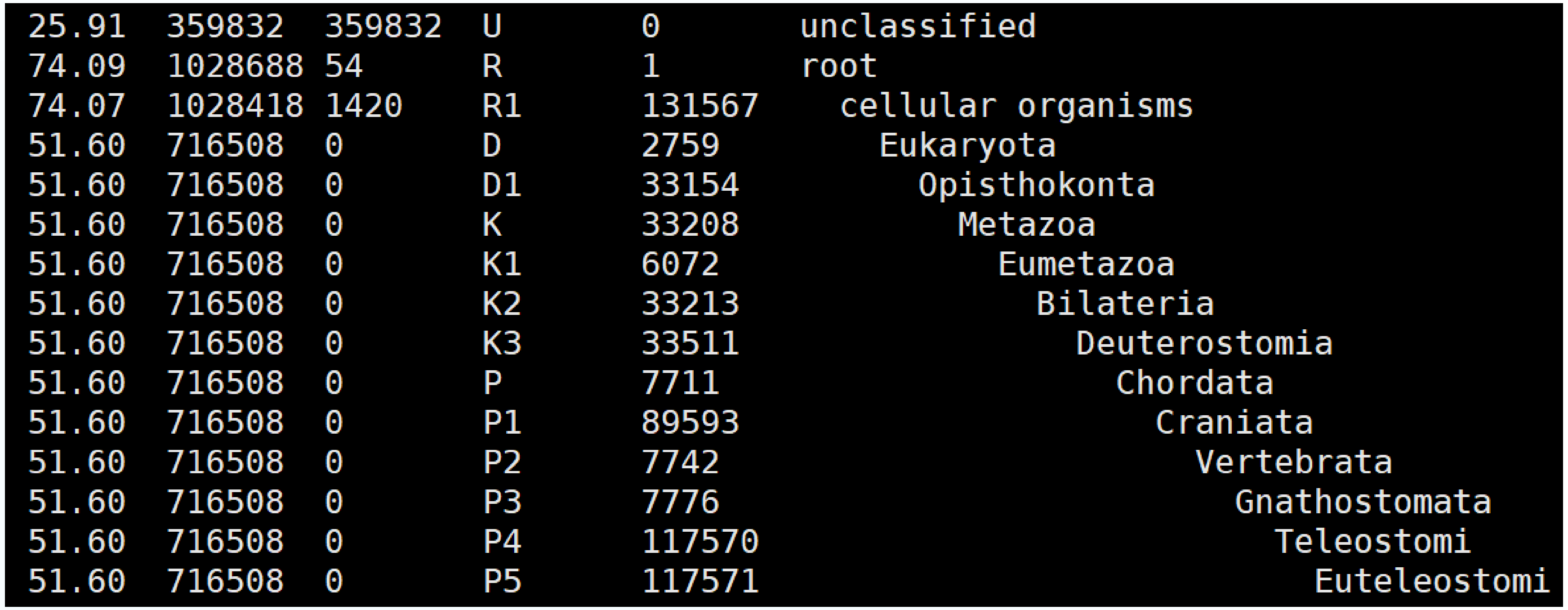
```
i=sample_name
kraken2     --db /path/to/kraken2_db/k2_standard/   --threads 24   --report
${i}_kraken2_report   --output ${i}_kraken_result   ${i}.fastq
```

- 合并多个样本的kraken2注释结果（KrakenTools）

```
python complete_kreports.py -r 1.KREPORT 2.KREPORT -o COMBINED.KREPORT
--no-headers --sample-names S1 S2
```

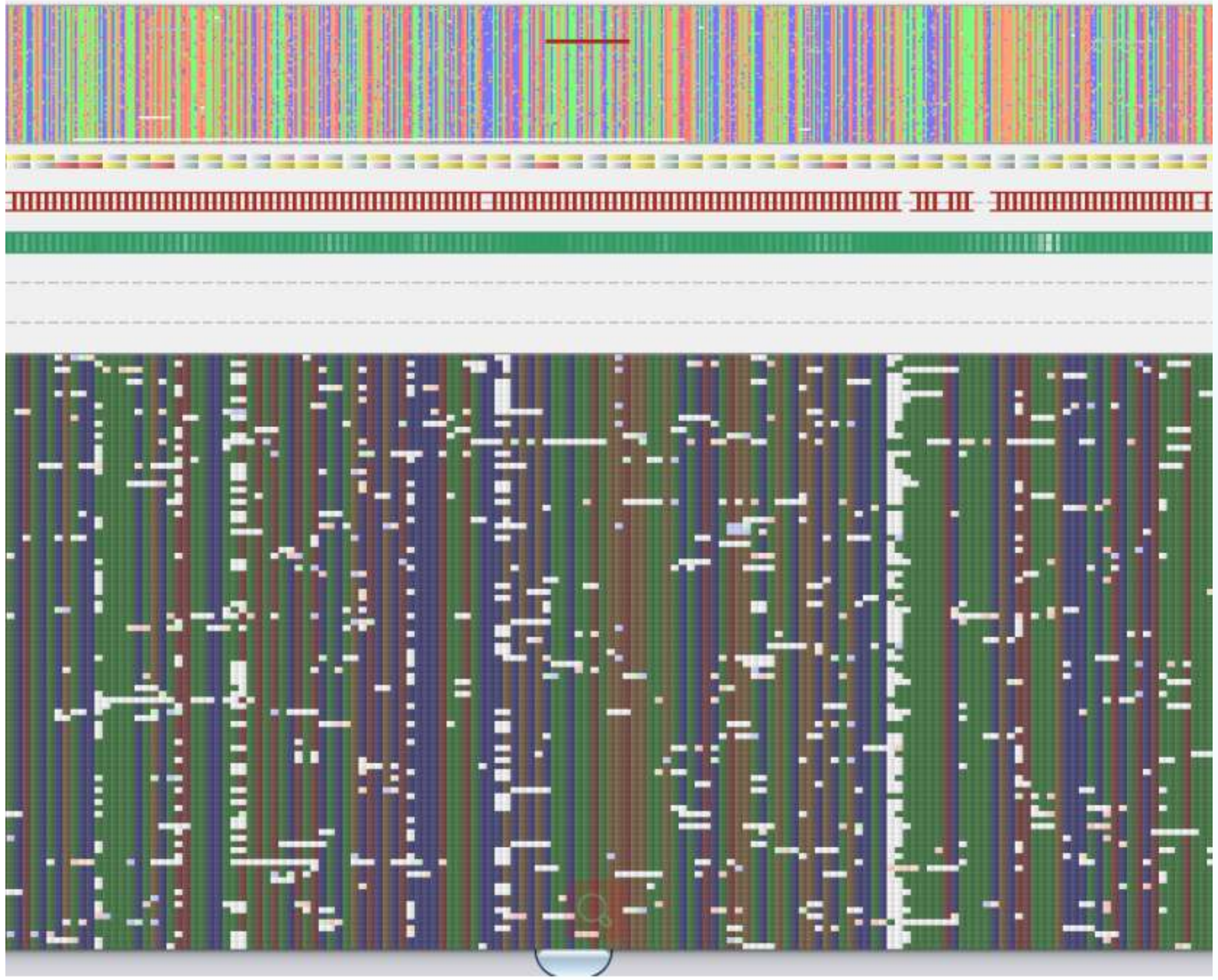
kraken2：结果展示

1. 物种所占百分比
2. 覆盖到该物种分类 clade rooted 的片段数
3. 比对到该物种分类的片段数
4. 物种分类等级代码：(U)nclassified，(R)oot,, (D)omain, (K)ingdom, (P)hylum, (C)lass, (O)rder, (F)amily, (G)enus, or (S)pecies.
5. NCBI 物种分类 ID
6. 微生物科学命名



```
25.91   359832  359832  U       0       unclassified
74.09   1028688 54      R       1       root
74.07   1028418 1420    R1      131567    cellular organisms
51.60   716508  0       D       2759        Eukaryota
51.60   716508  0       D1      33154         Opisthokonta
51.60   716508  0       K       33208           Metazoa
51.60   716508  0       K1      6072              Eumetazoa
51.60   716508  0       K2      33213               Bilateria
51.60   716508  0       K3      33511                 Deuterostomia
51.60   716508  0       P       7711                    Chordata
51.60   716508  0       P1      89593                     Craniata
51.60   716508  0       P2      7742                        Vertebrata
51.60   716508  0       P3      7776                          Gnathostomata
51.60   716508  0       P4      117570                          Teleostomi
51.60   716508  0       P5      117571                            Euteleostomi
```

直接从纳米孔长读测序数据中鉴定及定量功能基因



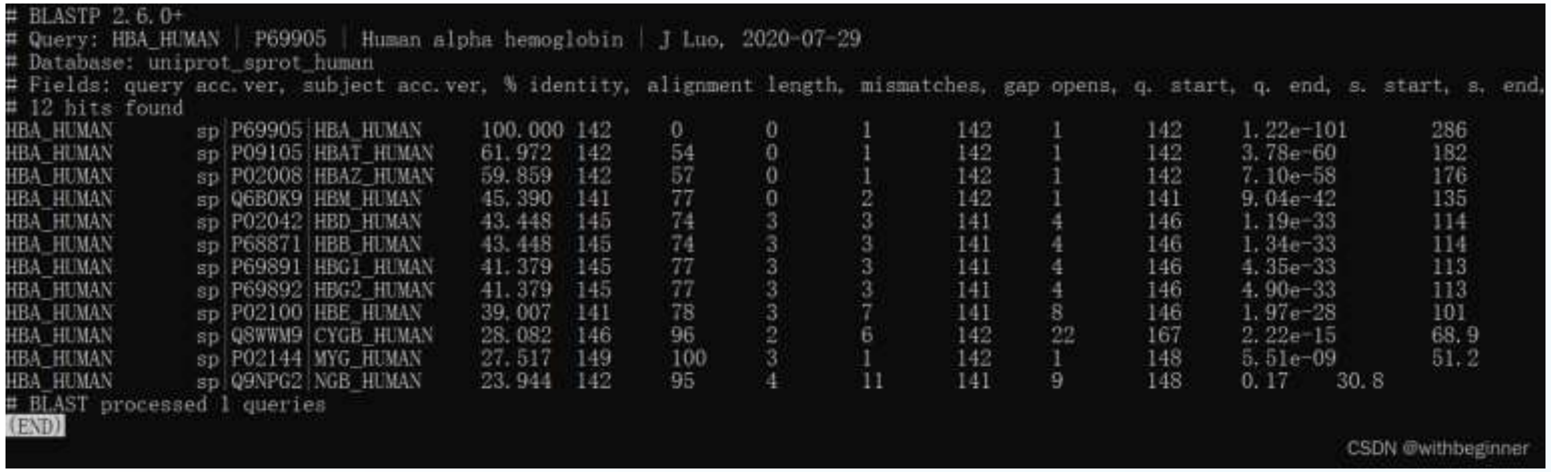
纳米孔测序的序列特征：
1、存在单碱基错误；
2、存在小的插入缺失错误，主要是缺失；
3、测序错误分布不随机。

无法直接从原始测序reads进行基因预测

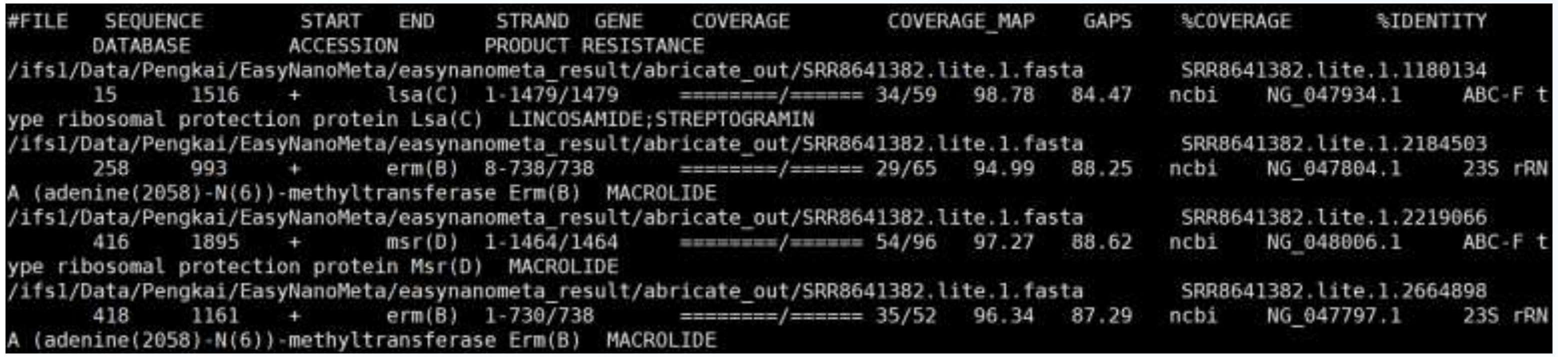
可以通过同源比对的方式进行功能基因鉴定---blastn

常见数据库：CARD，VFDB，CAZy，BacMet 等。

abricate：挖掘三代宏基因组数据功能基因



```
# BLASTP 2.6.0+
# Query: HBA_HUMAN | P69905 | Human alpha hemoglobin | J Luo, 2020-07-29
# Database: uniprot_sprot_human
# Fields: query acc.ver, subject acc.ver, % identity, alignment length, mismatches, gap opens, q. start, q. end, s. start, s. end,
# 12 hits found
HBA_HUMAN       sp|P69905|HBA_HUMAN     100.000 142     0       0       1       142     1       142     1.22e-101       286
HBA_HUMAN       sp|P09105|HBAT_HUMAN    61.972  142     54      0       1       142     1       142     3.78e-60        182
HBA_HUMAN       sp|P02008|HBAZ_HUMAN    59.859  142     57      0       1       142     1       142     7.10e-58        176
HBA_HUMAN       sp|Q6B0K9|HBM_HUMAN     45.390  141     77      0       2       142     1       141     9.04e-42        135
HBA_HUMAN       sp|P02042|HBD_HUMAN     43.448  145     74      3       3       141     4       146     1.19e-33        114
HBA_HUMAN       sp|P68871|HBB_HUMAN     43.448  145     74      3       3       141     4       146     1.34e-33        114
HBA_HUMAN       sp|P69891|HBG1_HUMAN    41.379  145     77      3       3       141     4       146     4.35e-33        113
HBA_HUMAN       sp|P69892|HBG2_HUMAN    41.379  145     77      3       3       141     4       146     4.90e-33        113
HBA_HUMAN       sp|P02100|HBE_HUMAN     39.007  141     78      3       7       141     8       146     1.97e-28        101
HBA_HUMAN       sp|Q8WWM9|CYGB_HUMAN    28.082  146     96      2       6       142     22      167     2.22e-15        68.9
HBA_HUMAN       sp|P02144|MYG_HUMAN     27.517  149     100     3       1       142     1       148     5.51e-09        51.2
HBA_HUMAN       sp|Q9NPG2|NGB_HUMAN     23.944  142     95      4       11      141     9       148     0.17    30.8
# BLAST processed 1 queries
(END)
```



```
#FILE   SEQUENCE        START   END     STRAND  GENE    COVERAGE        COVERAGE_MAP    GAPS    %COVERAGE       %IDENTITY
        DATABASE        ACCESSION       PRODUCT RESISTANCE
/ifs1/Data/Pengkai/EasyNanoMeta/easynanometa_result/abricate_out/SRR8641382.lite.1.fasta        SRR8641382.lite.1.1180134
        15      1516    +       lsa(C)  1-1479/1479     ========/====== 34/59   98.78   84.47   ncbi    NG_047934.1     ABC-F t
ype ribosomal protection protein Lsa(C)  LINCOSAMIDE;STREPTOGRAMIN
/ifs1/Data/Pengkai/EasyNanoMeta/easynanometa_result/abricate_out/SRR8641382.lite.1.fasta        SRR8641382.lite.1.2184503
        258     993     +       erm(B)  8-738/738       ========/====== 29/65   94.99   88.25   ncbi    NG_047804.1     23S rRN
A (adenine(2058)-N(6))-methyltransferase Erm(B)  MACROLIDE
/ifs1/Data/Pengkai/EasyNanoMeta/easynanometa_result/abricate_out/SRR8641382.lite.1.fasta        SRR8641382.lite.1.2219066
        416     1895    +       msr(D)  1-1464/1464     ========/====== 54/96   97.27   88.62   ncbi    NG_048006.1     ABC-F t
ype ribosomal protection protein Msr(D)  MACROLIDE
/ifs1/Data/Pengkai/EasyNanoMeta/easynanometa_result/abricate_out/SRR8641382.lite.1.fasta        SRR8641382.lite.1.2664898
        418     1161    +       erm(B)  1-730/738       ========/====== 35/52   96.34   87.29   ncbi    NG_047797.1     23S rRN
A (adenine(2058)-N(6))-methyltransferase Erm(B)  MACROLIDE
```

简化了blastn的输出结果，使匹配结果唯一，增加了结果的可读性。

找到约 4,840 条结果 （用时0.10秒）

abricate：软件安装

- 软件安装（需指定版本，否则安装老版本软件）

```
conda create -n abricate
conda activate abricate
conda install -y -c bioconda abricate=1.0.1
```

abricate：数据库配置

- 查看当前的数据库

```
abricate --list
```

- 更新已有的数据库

```
abricate-get_db --db ncbi
abricate-get_db --db card
```

- 创建用户自定义数据库

```
cd /path/to/abricate/db
mkdir ${i}db
cp /your/database/database.fasta your_database_name/sequences
makeblastdb -in sequences -title your_database_name -dbtype nucl -hash_index
```

abricate：功能基因定量

- 每Gb测序数据中的基因拷贝数：

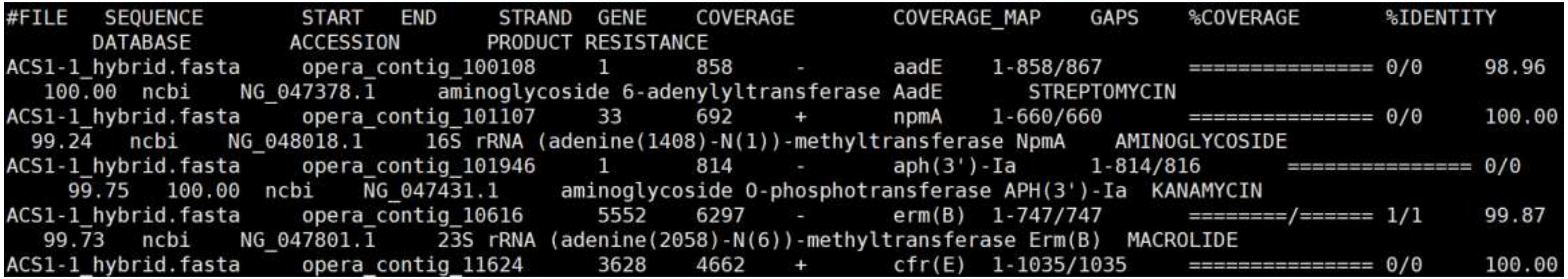
$$\mathrm{ARG}_i \mathrm{abundance} \cdot (\mathrm{gc/Gb}) = \frac{\sum_1^m \frac{\mathrm{Alighment}_{\mathrm{end}} - \mathrm{Aligment}_{\mathrm{start}}}{\mathrm{Length}_{\mathrm{reference}}(\mathrm{bp})}}{\frac{\sum_1^n \mathrm{Length}_{\mathrm{read}}(\mathrm{bp})}{10^9 \mathrm{bp/Gb}}}$$

- Alighment end与start为对齐到参考基因的片段长度
- Length reference为参考基因的片段长读
- 对这些比值进行求和
- 分母为测序数据量

abricate：使用案例

- 使用NCBI的AMRfinder数据库进行耐药基因鉴定
- 默认参数的coverage和identity均为80，宏基因组数据需根据情况进行调整，多数文献使用 --coverage 40 --identity 70

```
conda activate abricate
i=sample_name
abricate --db ncbi --mincov 40 --minid 70 -t 24 ${i}.fasta > ${i}_ncbi_result
```



```
#FILE    SEQUENCE        START    END      STRAND   GENE     COVERAGE        COVERAGE_MAP     GAPS     %COVERAGE        %IDENTITY
       DATABASE         ACCESSION        PRODUCT RESISTANCE
ACS1-1_hybrid.fasta     opera_contig_100108      1        858      -        aadE     1-858/867       =============== 0/0      98.96
   100.00  ncbi     NG_047378.1     aminoglycoside 6-adenylyltransferase AadE        STREPTOMYCIN
ACS1-1_hybrid.fasta     opera_contig_101107      33       692      +        npmA     1-660/660       =============== 0/0      100.00
  99.24    ncbi     NG_048018.1     16S rRNA (adenine(1408)-N(1))-methyltransferase NpmA    AMINOGLYCOSIDE
ACS1-1_hybrid.fasta     opera_contig_101946      1        814      -        aph(3')-Ia       1-814/816       =============== 0/0
    99.75   100.00  ncbi     NG_047431.1     aminoglycoside O-phosphotransferase APH(3')-Ia  KANAMYCIN
ACS1-1_hybrid.fasta     opera_contig_10616       5552     6297     -        erm(B)   1-747/747       ========/====== 1/1      99.87
   99.73   ncbi     NG_047801.1     23S rRNA (adenine(2058)-N(6))-methyltransferase Erm(B)  MACROLIDE
ACS1-1_hybrid.fasta     opera_contig_11624       3628     4662     +        cfr(E)   1-1035/1035     =============== 0/0      100.00
```

abricate：功能基因定量

- 使用我们的脚本对耐药基因进行定量

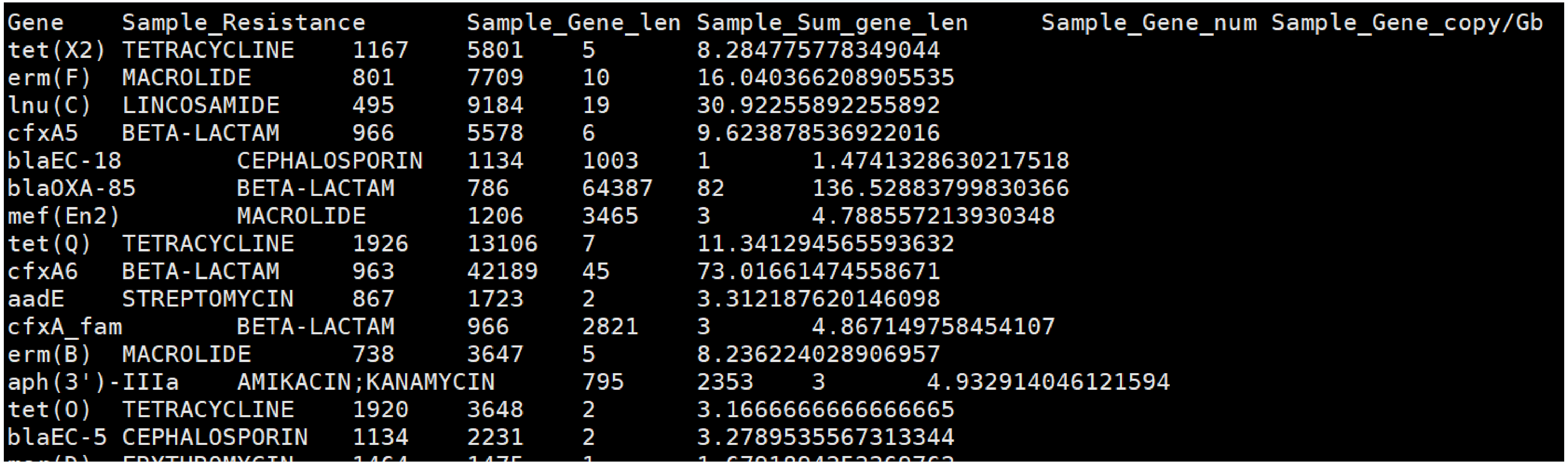
https://github.com/P-kai/EasyNanoMeta/tree/main/Python%20scripts%20for%20%20data%20analysis

```
python abundance_calculate.py --help
usage: abundance_calculate.py [-h] [--i I] [--data_size DATA_SIZE] [--title TITLE] [--p P] [--output OUTPUT]
options:
 -h, --help show this help message and exit
 --i I, -i I Input data.
 --data_size DATA_SIZE, -d DATA_SIZE
 --data_size, -d
 --title TITLE
 --p P, -p P The prefix of result.
 --output OUTPUT, -o OUTPUT    Output direction.
```

abricate：功能基因定量

- 使用我们的脚本对耐药基因进行定量

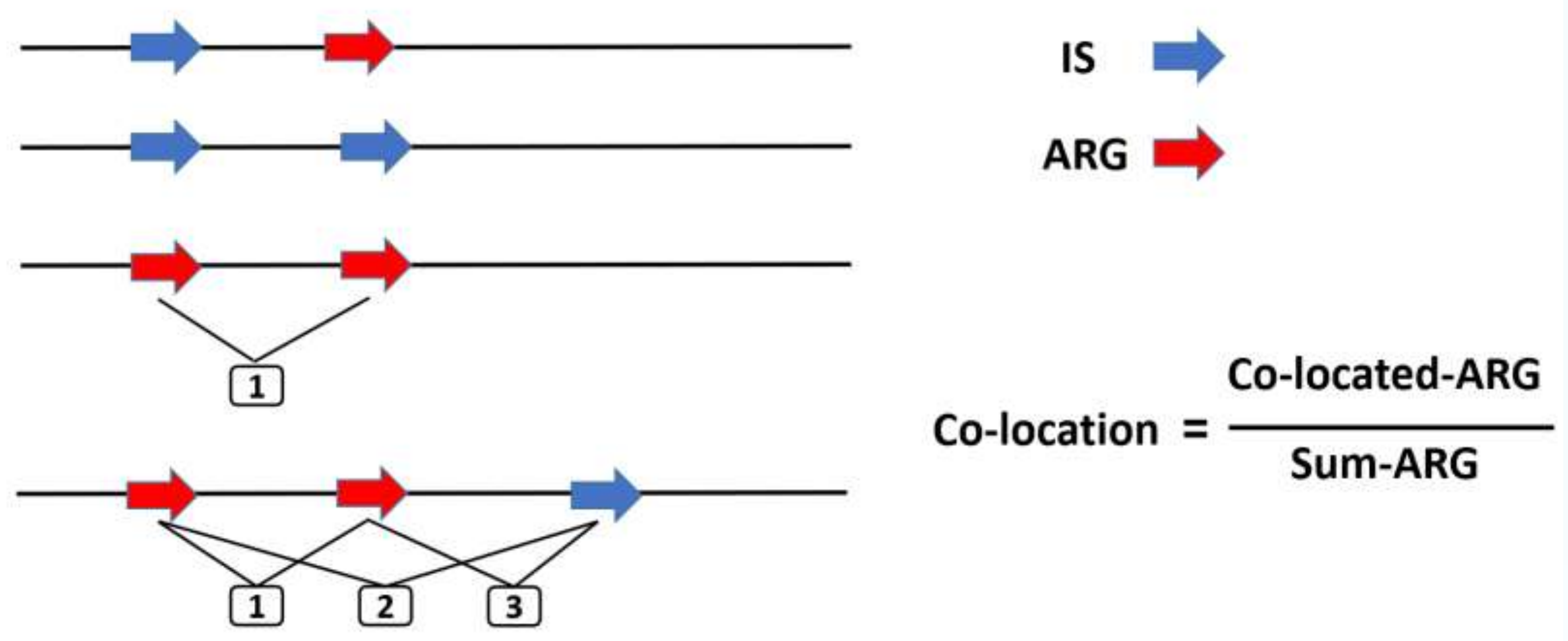
i=sample_name
seqkit stats sample_name
python abundance_calculate.py -i ${i}_ncbi_result --data_size 0.75 --title {i} --p {i} -o {i}_quant



```
Gene     Sample_Resistance       Sample_Gene_len Sample_Sum_gene_len     Sample_Gene_num Sample_Gene_copy/Gb
tet(X2) TETRACYCLINE    1167    5801    5       8.284775778349044
erm(F)  MACROLIDE       801     7709    10      16.040366208905535
lnu(C)  LINCOSAMIDE     495     9184    19      30.92255892255892
cfxA5   BETA-LACTAM     966     5578    6       9.623878536922016
blaEC-18        CEPHALOSPORIN   1134    1003    1       1.4741328630217518
blaOXA-85       BETA-LACTAM     786     64387   82      136.52883799830366
mef(En2)        MACROLIDE       1206    3465    3       4.788557213930348
tet(Q)  TETRACYCLINE    1926    13106   7       11.341294565593632
cfxA6   BETA-LACTAM     963     42189   45      73.01661474558671
aadE    STREPTOMYCIN    867     1723    2       3.312187620146098
cfxA_fam        BETA-LACTAM     966     2821    3       4.867149758454107
erm(B)  MACROLIDE       738     3647    5       8.236224028906957
aph(3')-IIIa    AMIKACIN;KANAMYCIN      795     2353    3       4.932914046121594
tet(O)  TETRACYCLINE    1920    3648    2       3.1666666666666665
blaEC-5 CEPHALOSPORIN   1134    2231    2       3.2789535567313344
```

基于长读测序数据的基因共现分析

基于宏基因组中单分子测序序列分析不同耐药基因之间或耐药基因与插入序列之间的共整合模式



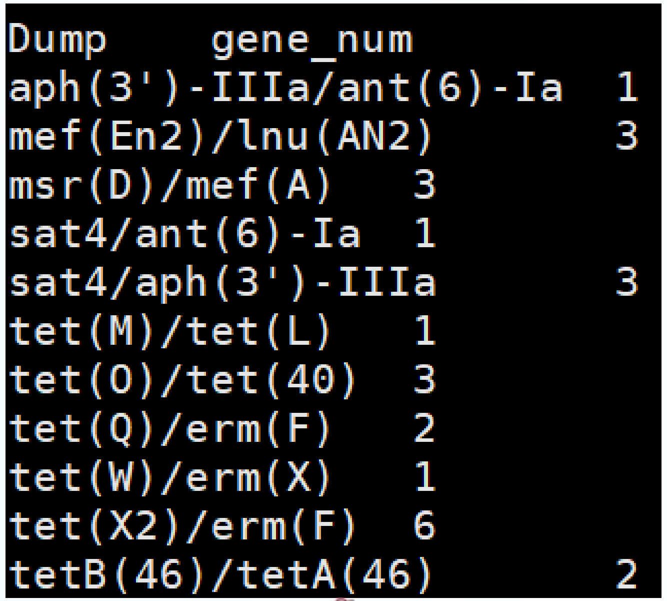
$$\text{Co-location} = \frac{\text{Co-located-ARG}}{\text{Sum-ARG}}$$

两个基因位于同一read上，则它们共存一次

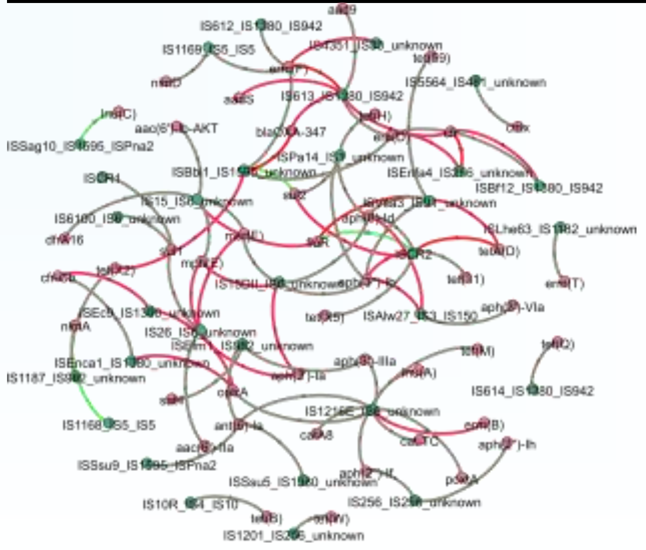
基于长读测序数据的基因共现分析

基于abricate脚本进行基因共现分析

```
python co-located.py --help
usage: co-located.py [-h] [--i I]
options:
 -h, --help show this help message and exit
 --i I

i=sample_name
python co-located.py --i ${i}_ncbi_result
```

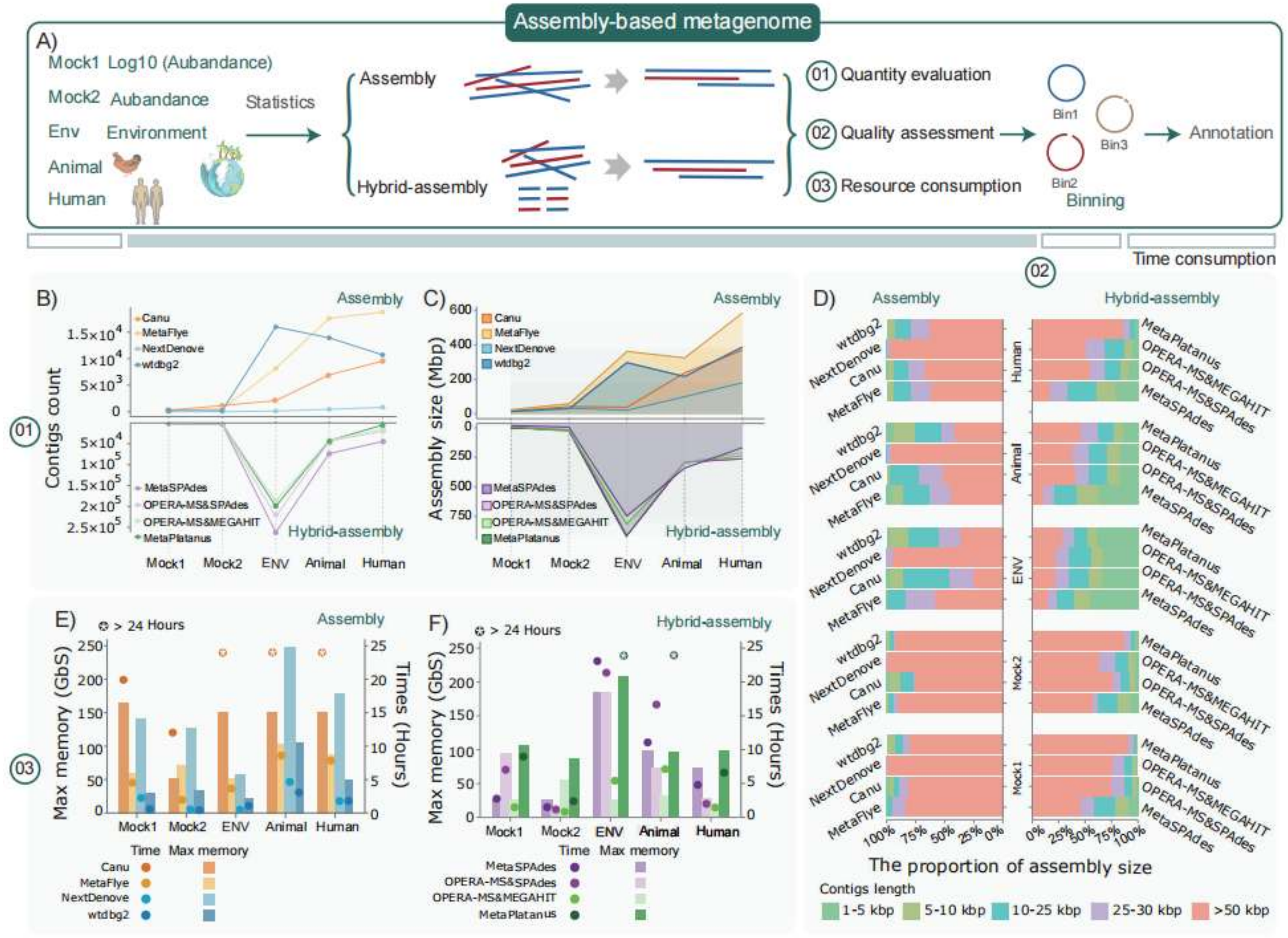


```
Dump    gene_num
aph(3')-IIIa/ant(6)-Ia  1
mef(En2)/lnu(AN2)       3
msr(D)/mef(A)   3
sat4/ant(6)-Ia  1
sat4/aph(3')-IIIa       3
tet(M)/tet(L)   1
tet(O)/tet(40)  3
tet(Q)/erm(F)   2
tet(W)/erm(X)   1
tet(X2)/erm(F)  6
tetB(46)/tetA(46)       2
```



5. 组装、评估和纠错

不同三代长读宏基因组组装软件性能评测



MetaFlye在组装结果、计算效率等方面的综合表现最佳。

metaflye：三代长读宏基因组组装最佳软件

安装MetaFlye

软件下载及解压

```
cd ~/tools
wget https://github.com/fenderglass/Flye/archive/refs/tags/2.9.2.tar.gz
tar -zxvf Flye-2.9.2.tar.gz
```

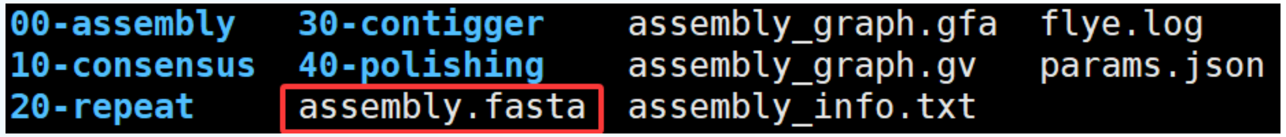
查看软件版本，版本：2.9.2-b1786

```
~/tools/Flye-2.9.2/bin/flye --version
```

metaflye：软件使用

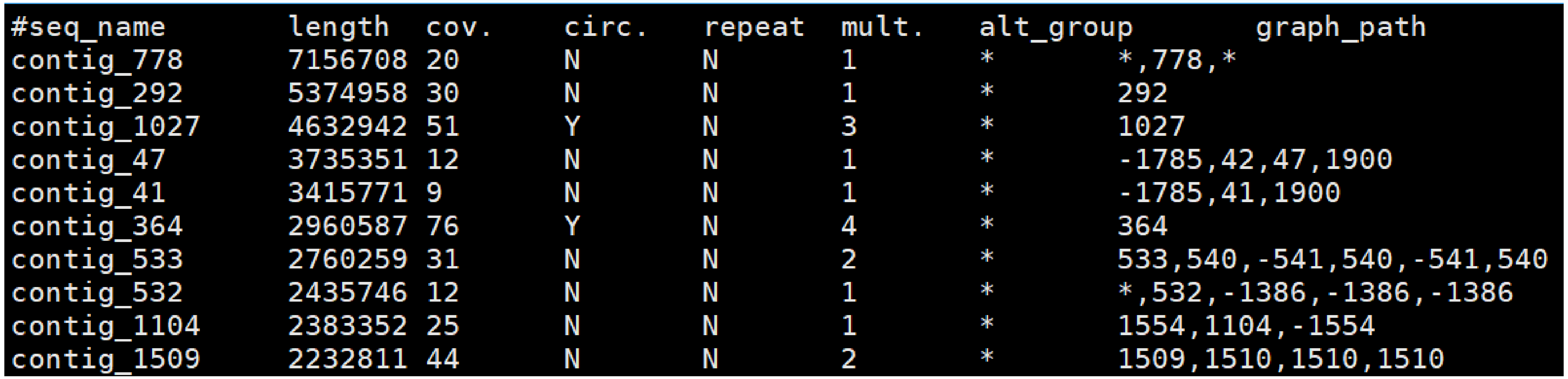
利用MetaFlye对测序数据进行组装

i=sample_name
~/path/to/Flye-2.9.2/bin/flye --meta --nano-raw ${i}.fasta --threads 24 --out-dir ${i}_flye



```
00-assembly    30-contigger     assembly_graph.gfa  flye.log
10-consensus   40-polishing     assembly_graph.gv   params.json
20-repeat      assembly.fasta   assembly_info.txt
```

--meta 对宏基因组进行组装
--nano-raw 纳米孔测序原始数据



```
#seq_name       length   cov.    circ.   repeat  mult.   alt_group       graph_path
contig_778      7156708 20       N       N       1       *       *,778,*
contig_292      5374958 30       N       N       1       *       292
contig_1027     4632942 51       Y       N       3       *       1027
contig_47       3735351 12       N       N       1       *       -1785,42,47,1900
contig_41       3415771 9        N       N       1       *       -1785,41,1900
contig_364      2960587 76       Y       N       4       *       364
contig_533      2760259 31       N       N       2       *       533,540,-541,540,-541,540
contig_532      2435746 12       N       N       1       *       *,532,-1386,-1386,-1386
contig_1104     2383352 25       N       N       1       *       1554,1104,-1554
contig_1509     2232811 44       N       N       2       *       1509,1510,1510,1510
```

quast、seqkit：评估组装结果

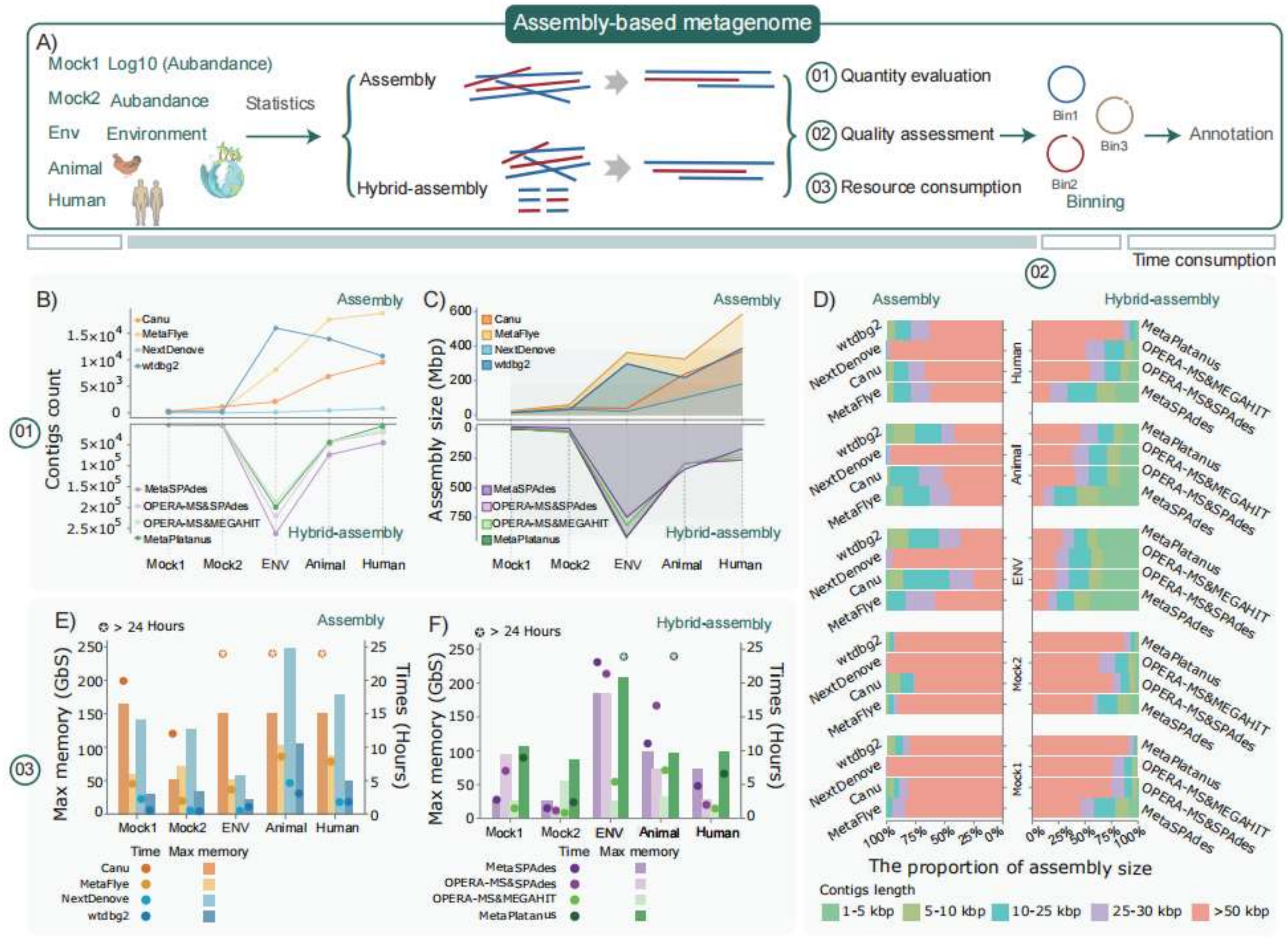
- 使用quast统计组装结果，包括contigs长度分布、N50、GC含量等参数
- 安装

```
wget https://github.com/ablab/quast/archive/refs/tags/quast_5.3.0.tar.gz
tar -zxvf quast_5.3.0.tar.gz

i=sample_name
quast.py ${i}.fa -o result/metaflye/quast
```

- 使用seqkit快速统计组装结果基础信息

```
seqkit stats assembly.fasta
```

长短读宏基因组数据混合组装



基于metaSPAdes的OPERA-MS组装效果更好。

基于MEGAHIT的OPERA-MS组装效率更高。

Peng, et al. ***Science Bulletin*** 2025. https://doi.org/10.1016/j.scib.2025.03.044

OPERA-MS_metaSPAdes进行长短读混合组装

- 软件安装：
- 使用conda配置软件安装单独环境，安装软件依赖的perl模块

```
conda create -n operams python=3.9
conda activate operams
```

- 在conda环境中安装依赖的perl模块

```
conda install -c conda-forge perl-app-cpanminus
conda install -c compbiocore perl-switch perl==5.26.2
conda install -c bioconda perl-file-which perl-statistics-basic perl-statistics-r
```

- 软件下载及编辑

```
cd ~/tools
git clone https://github.com/CSB5/OPERA-MS.git
cd OPERA-MS
make
perl OPERA-MS.pl check-dependency
```

OPERA-MS安装可能遇到的问题以及数据库配置

可能遇到问题 “Can't locate Switch.pm”

解决：
寻找当前用户目录下有没有Switch.pm模块 find ~/ -name "Switch.pm"

将找到的模块写入perl路径中
例如： export PERL5LIB=~/perl5/lib/perl5/

- 完成软件安装后，配置OPERA-MS软件数据库

perl OPERA-MS.pl install-db

MetaSPAdes安装

- 直接下载预编译的软件安装包，解压后使用

```
cd ~/tools
wget https://github.com/ablab/spades/releases/download/v3.15.5/SPAdes-3.15.5-Linux.tar.gz
tar -zxvf SPAdes-3.15.5-Linux.tar.gz
```

OPERA-MS_metaSPAdes：软件使用

- 激活OPERA-MS依赖环境，使用软件进行组装

```
conda activate operams
perl ../OPERA-MS.pl  --short-read1 R1.fastq.gz  --short-read2 R2.fastq.gz  --short-read-assembler spades  --long-read long_read.fastq  --no-ref-clustering  --no-polishing  --num-processors 24  --out-dir OPERA-MS_metaSPAdes
```

- short-read1以及short-read2 对应短读测序的双端测序数据
- short-read-assembler 定义短读组装软件，默认megahit，需指定metaSPAdes
- long-read 为纳米孔测序数据
- no-polishing 跳过纠错步骤，这一步使用的pilon，特别耗时，且经常报错

OPERA-MS_metaSPAdes：软件使用

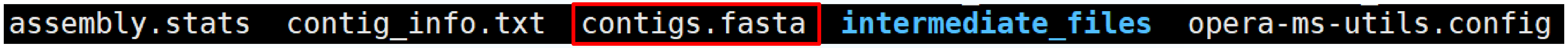
- 已完成二代组装，使用OPERA-MS进行混合组装

```
conda activate operams
perl ../OPERA-MS.pl  --contig-file metaSPAdes.fasta  --long-read long_read.fastq  --no-ref-clustering  --no-polishing  --num-processors 24  --out-dir OPERA-MS_metaSPAdes
```

- contig-file为二代组装结果
- long-read为纳米孔测序数据
- no-polishing跳过纠错步骤，这一步使用的pilon，特别耗时，且经常报错

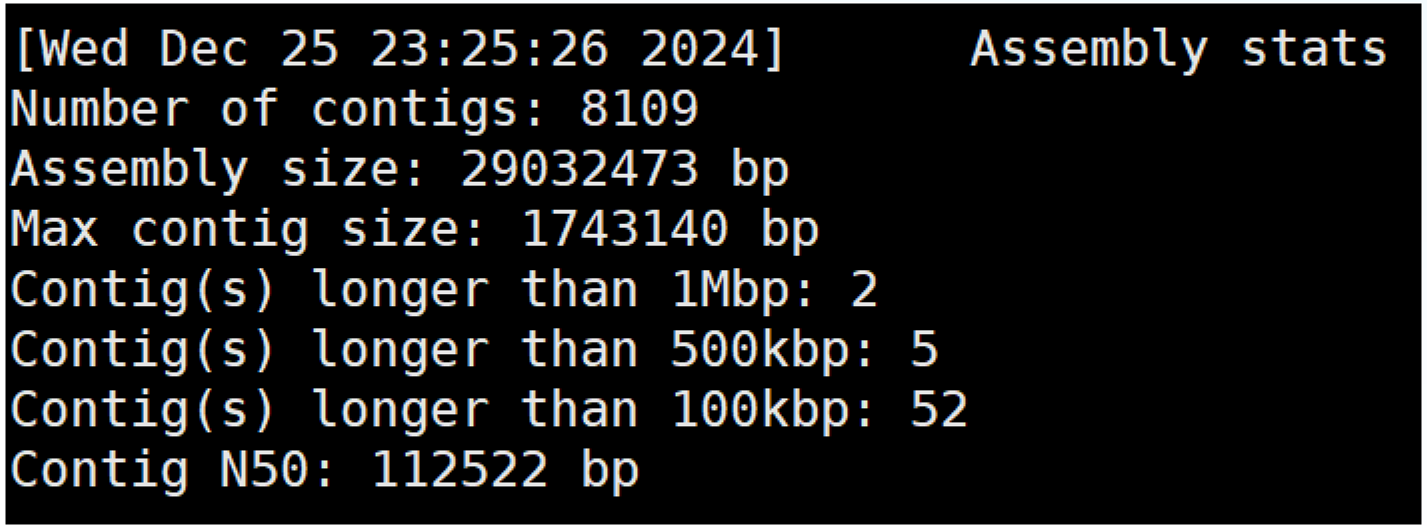
OPERA-MS_metaSPAdes：结果文件

软件输出文件夹：



```
assembly.stats  contig_info.txt  contigs.fasta  intermediate_files  opera-ms-utils.config
```

组装结果：



```
[Wed Dec 25 23:25:26 2024]      Assembly stats
Number of contigs: 8109
Assembly size: 29032473 bp
Max contig size: 1743140 bp
Contig(s) longer than 1Mbp: 2
Contig(s) longer than 500kbp: 5
Contig(s) longer than 100kbp: 52
Contig N50: 112522 bp
```

quast、seqkit：评估组装结果

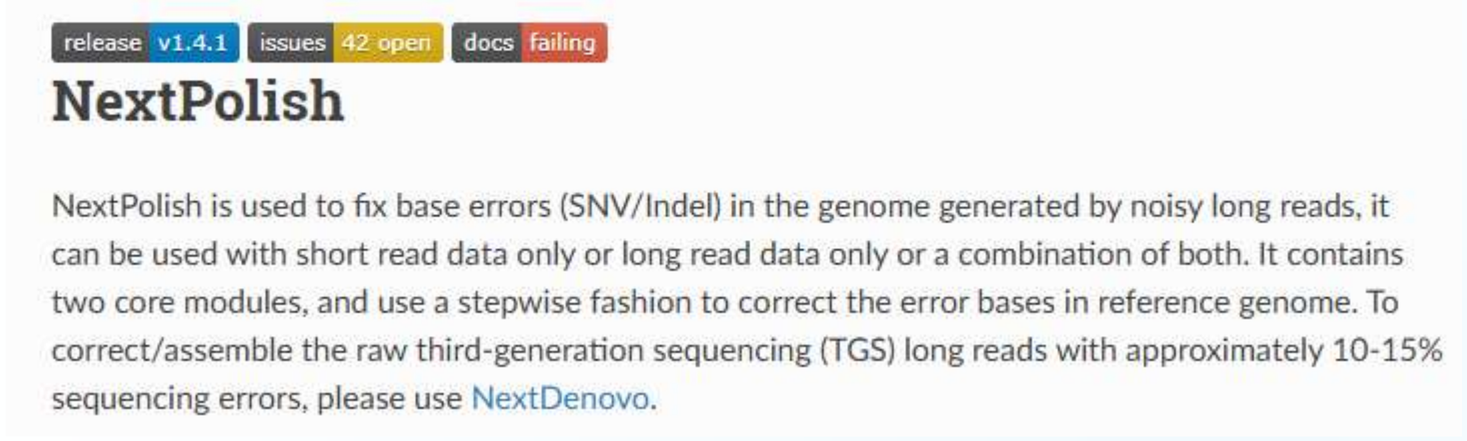
- 使用quast统计组装结果，包括contigs长度分布、N50、GC含量等参数
- 安装

```
wget https://github.com/ablab/quast/archive/refs/tags/quast_5.3.0.tar.gz
tar -zxvf quast_5.3.0.tar.gz

i=sample_name
quast.py ${i}.fa -o result/operams/quast
```

- 使用seqkit快速统计组装结果基础信息

```
seqkit stats assembly.fasta
```

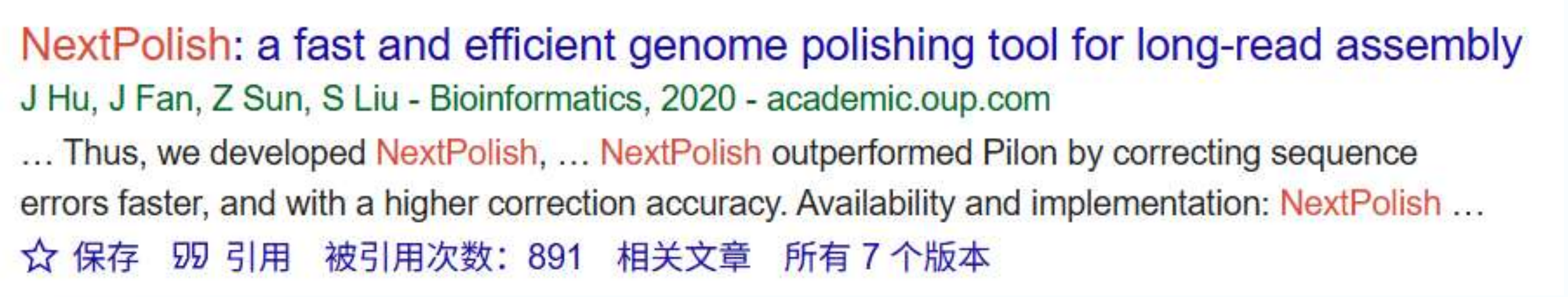
NextPolish：组装基因组纠错



release v1.4.1 issues 42 open docs failing

NextPolish

NextPolish is used to fix base errors (SNV/Indel) in the genome generated by noisy long reads, it can be used with short read data only or long read data only or a combination of both. It contains two core modules, and use a stepwise fashion to correct the error bases in reference genome. To correct/assemble the raw third-generation sequencing (TGS) long reads with approximately 10-15% sequencing errors, please use NextDenovo.



NextPolish: a fast and efficient genome polishing tool for long-read assembly
J Hu, J Fan, Z Sun, S Liu - Bioinformatics, 2020 - academic.oup.com
… Thus, we developed NextPolish, … NextPolish outperformed Pilon by correcting sequence errors faster, and with a higher correction accuracy. Availability and implementation: NextPolish …
☆ 保存 99 引用 被引用次数：891 相关文章 所有 7 个版本

https://nextpolish.readthedocs.io/en/latest/

NextPolish：软件安装

- 软件安装

```
cd ~/tools
wget
https://github.com/Nextomics/NextPolish/releases/latest/download/NextPolish.tgz
tar -vxzf NextPolish.tgz && cd NextPolish && make
```

- 安装软件依赖

```
pip install paralleltask
```

NextPolish：使用二代数据进行组装结果纠错

- 生成二代数据路径文件

```
ls reads1.fq reads2.fa.gz > sgs.fofn
```

- 编辑组装结果校准的可执行文件，在该文件中配置软件执行参数

```
vim run.cfg
```

- 执行组装结果校准程序

```
path/nextPolish run.cfg
```

```
[General]
job_type = local
job_prefix = nextPolish
task = best
rewrite = yes
rerun = 3
parallel_jobs = 6
multithread_jobs = 5
genome = ./raw.genome.fasta #组装结果文件
genome_size = auto
workdir = ./short-reads-polish
polish_options = -p 8

[sgs_option]
sgs_fofn = ./sgs.fofn
sgs_options = -max_depth 100 -bwa
```

NextPolish：使用二代及三代数据进行组装结果纠错

- 生成二代及三代数据路径文件

```
ls reads1.fq reads2.fa.gz > sgs.fofn
ls long_reads.fq > lgs.fofn
```

- 编辑组装结果校准的可执行文件，在该文件中配置软件执行参数

```
vim run.cfg
```

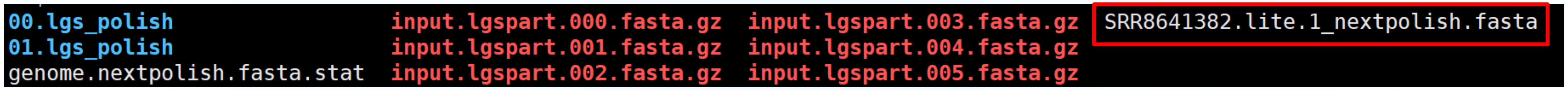
- 执行组装结果校准程序

```
path/nextPolish run.cfg
```

```
[General]
job_type = local
job_prefix = nextPolish
task = best
rewrite = yes
rerun = 3
parallel_jobs = 6
multithread_jobs = 5
genome = ./raw.genome.fasta #组装结果文件
genome_size = auto
workdir = ./short-reads-polish
polish_options = -p 8

[sgs_option]
sgs_fofn = ./sgs.fofn
sgs_options = -max_depth 100 -bwa

[lgs_option]
lgs_fofn = ./lgs.fofn
lgs_options = -min_read_len 1k -max_depth 100
lgs_minimap2_options = -x map-ont
```

NextPolish：结果文件

软件输出文件夹：



```
00.lgs_polish                  input.lgspart.000.fasta.gz  input.lgspart.003.fasta.gz  SRR8641382.lite.1_nextpolish.fasta
01.lgs_polish                  input.lgspart.001.fasta.gz  input.lgspart.004.fasta.gz
genome.nextpolish.fasta.stat   input.lgspart.002.fasta.gz  input.lgspart.005.fasta.gz
```

6. 分箱和物种注释

eggnog-mapper：组装宏基因组功能注释

- 软件安装

```
conda create -n eggnog-mapper
conda activate eggnog-mapper
conda install eggnogmapper
```

- 数据库下载

```
mkdir ~/db/eggnog-mapper && cd ~/db/eggnog-mapper
download_eggnog_data.py --data_dir ~/db/eggnog-mapper -y -f -P -M -H -d taxid
```

eggnog-mapper：组装宏基因组功能注释

- 设置数据库位置

```
export EGGNOG_DATA_DIR=/your/database/path/eggnog-mapper/
```

- 对组装结果进行功能注释

```
i=sample_name
conda activate eggnog-mapper
emapper.py -i ${i}.fasta  -o ${i}   --itype metagenome --cpu 24
```

使用KEGG数据库注释组装宏基因组

构建细菌KEGG本地数据库
详细的步骤参见：
https://github.com/P-kai/EasyNanoMeta/blob/main/install.sh

使用KEGG数据库注释组装宏基因组

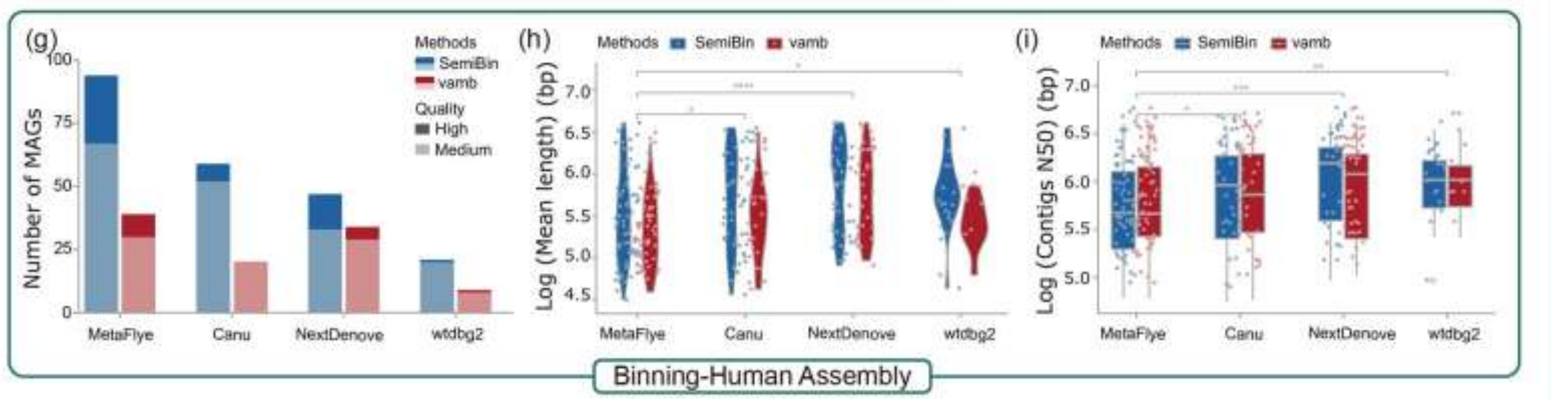
使用prodigal进行宏基因组组装结果的基因预测

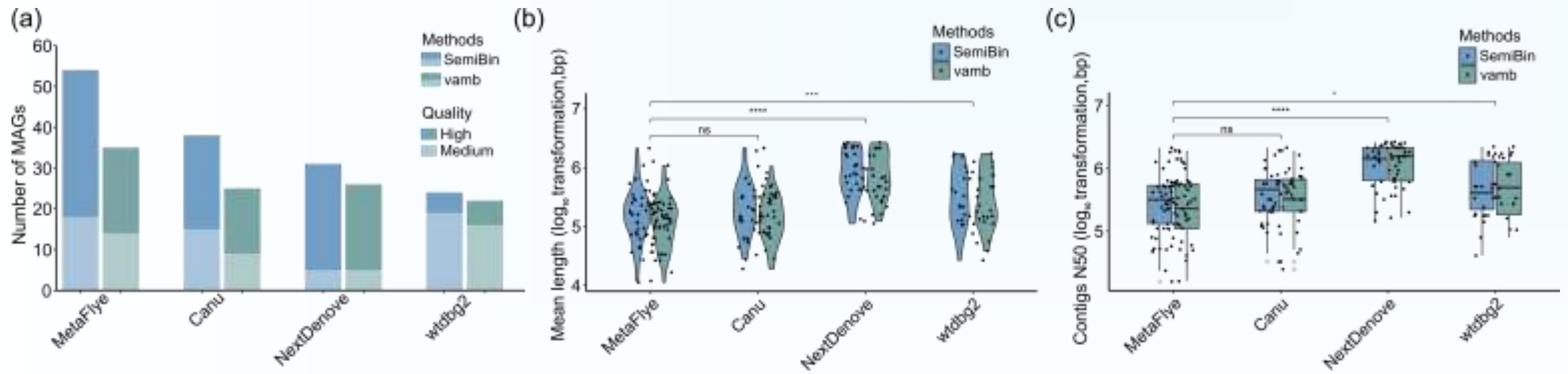
```
i=sample_name
prodigal -i ${i}.fasta -f gff -o ${i}_gene.gff3 -p meta -d ${i}_gene.fna -a
${i}_gene.faa
```

使用diamond进行基因功能比对

```
diamond blastx -q ${i}_gene.fna -d kb_refseq.dmnd  --max-hsps 1 --max-target-
seqs 1 --sensitive --outfmt 6 --evalue 1e-5 -p 16  -o kegg_${i}_match.out
```

长读组装宏基因组分箱--软件选择





目前仅有SemiBin和vamb可进行三代组装宏基因组分箱，总体而言，SemiBin性能显著优于vamb。

[Science Bulletin | 扬大王志强/基因组所刘永鑫开发纳米孔宏基因组分析流程EasyNanoMeta]

Peng, et al. ***Science Bulletin*** 2025. https://doi.org/10.1016/j.scib.2025.03.044

SemiBin：长读组装宏基因组分箱

- 安装SemiBin
- 使用conda创建单独的环境进行软件安装

```
conda create -n SemiBin
conda activate SemiBin
conda install -c conda-forge -c bioconda semibin
```

- 使用conda的package进行软件安装

```
cd ~/tools
wget -c --no-check-certificate --no-proxy
https://figshare.com/ndownloader/files/45563634 -O SemiBin.tar.gz
mkdir ~/miniconda3/envs/SemiBin/
tar -xzvf SemiBin.tar.gz -C ~/miniconda3/envs/SemiBin/
conda activate SemiBin
conda unpack
```

SemiBin：软件使用

- 软件使用
- 首先对组装基因组进行索引创建，获取排序的bam文件

```
i=sample_name
minimap2 -d catalogue.mmi ${i}.fasta
```

- 比对获取bam文件,raw_data.fq.gz为测序的原始数据

```
minimap2 -t 8 -N 5 -ax map-ont catalogue.mmi --split-prefix
mmsplit ../raw_data.fq.gz | samtools view -F 3584 -b --threads 8 > ${i}.bam
```

- 对bam文件进行排序

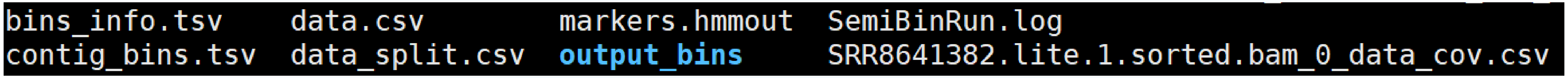
```
samtools sort -@ 10 ${i}.bam > ${i}.sorted.bam
```

- 使用SemiBin运行bin

```
SemiBin single_easy_bin -i ${i}.fasta  --sequencing-type long_read -b
${i}.sorted.bam -o bin_output --environment global
```

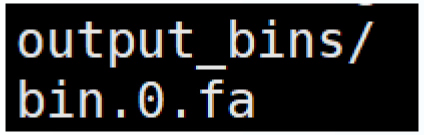
SemiBin：结果展示

软件输出文件夹：



```
bins_info.tsv      data.csv         markers.hmmout   SemiBinRun.log
contig_bins.tsv    data_split.csv   output_bins      SRR8641382.lite.1.sorted.bam_0_data_cov.csv
```

bins储存位置：



```
output_bins/
bin.0.fa
```

metawrap：混合组装宏基因组分箱

- 使用conda进行软件安装（一般不成功，原因未知，该软件很久不维护了）

```
conda create -y -n metawrap-env python=2.7
conda activate metawrap-env

conda config --add channels defaults
conda config --add channels conda-forge
conda config --add channels bioconda
conda config --add channels ursky

conda install -y mamba
mamba install --only-deps -c ursky metawrap-mg
```

metawrap：软件安装

- 使用conda package进行软件安装

```
cd ~/tools
```

- 下载打包的metawrap conda package

```
wget -c --no-check-certificate --no-proxy
https://figshare.com/ndownloader/files/45651492 -O metawrap.tar.gz
mkdir ~/miniconda3/envs/metawrap/
tar -xzvf metawrap.tar.gz -C ~/miniconda3/envs/metawrap/
conda activate metawrap
conda unpack
```

metawrap：软件使用

- 使用metawrap里面的分箱模式进行二三代宏基因组组装数据分箱
- 运行分箱，原始数据为二代数据，格式必须为*_1.fastq；*_2.fastq

```
i=sample_name
metawrap binning --metabat2 --maxbin2 --concoct -t 48 --run-checkm -a ${i}.fa -o bin ${i}_clean_1.fastq ${i}_clean_2.fastq
```

checkm：质控宏基因组bins

- 安装checkm
- 也可使用conda创建环境单独安装checkm
- 创建python=3.9的conda环境

```
conda create -n checkm python=3.9
conda activate checkm
```

- 使用pip3安装checkm及其依赖环境

```
pip3 install numpy
pip3 install matplotlib
pip3 install pysam
pip3 install checkm-genome
```

checkm：数据库配置

- 配置checkm数据库

```
cd ~/db/
mkdir checmk_db && cd checmk_db
wget -c
https://data.ace.uq.edu.au/public/CheckM_databases/checkm_data_2015_01_16
.tar.gz
tar -zxvf checkm_data_2015_01_16.tar.gz
```

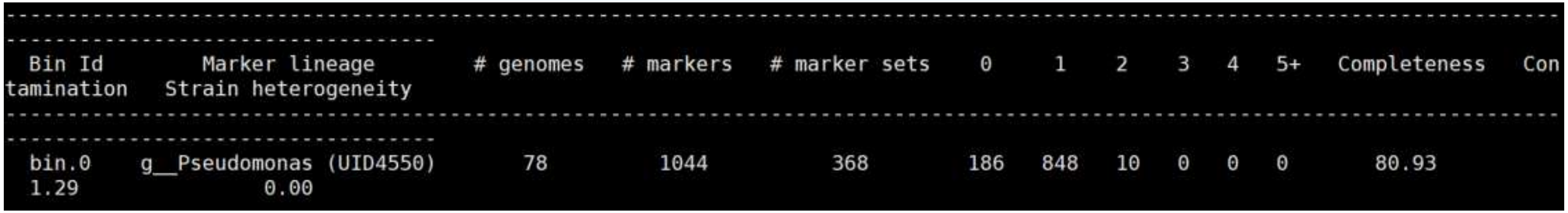
- 设置checkm数据库路径

```
checkm data setRoot $PATH/checkm_data
```

checkm：软件使用

- checkm使用：

checkm lineage_wf <bin folder>   <output folder>



```
------------------------------------------------------------------------------------------------------------------------------
----------------------------------
  Bin Id          Marker lineage          # genomes   # markers   # marker sets    0     1     2    3    4    5+   Completeness   Con
tamination   Strain heterogeneity
------------------------------------------------------------------------------------------------------------------------------
----------------------------------
  bin.0     g__Pseudomonas (UID4550)        78          1044          368         186   848   10    0    0    0       80.93
  1.29                   0.00
```

checkm2：质控宏基因组bins

- 安装checkm2
- 使用conda安装checkm2
- 创建checkm2环境，注意python版本为3.8，否则可能安装失败

```
conda create -n checkm2 python=3.8
conda activate checkm2
```

- 使用mamba安装

```
mamba install -c bioconda -c conda-forge checkm2
```

checkm2：质控宏基因组bins

- 使用conda package安装checkm2
- 下载conda package包

```
cd ~/tools
wget -c --no-check-certificate --no-proxy
https://figshare.com/ndownloader/files/45700833 -O checkm2.tar.gz

mkdir ~/miniconda3/envs/checkm2/
tar -xzvf checkm2.tar.gz -C ~/miniconda3/envs/checkm2/
conda activate checkm2
conda unpack
```

checkm2：数据库配置

- 直接使用checkm2脚本进行数据库下载（经常失败，推荐使用wget）

```
checkm2 database --download
```

- 使用wget进行数据库下载

```
mkdir ~/db/checkm2 && cd ~/db/checkm2
wget https://zenodo.org/record/5571251/files/checkm2_database.tar.gz
```

- 解压数据库

```
tar -zxvf checkm2_database.tar.gz
```

checkm2：软件使用

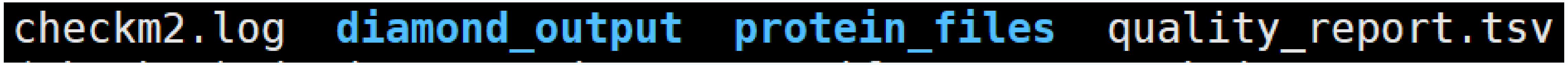
- 使用checkm2质控bins

```
checkm2 predict --threads 24   --input bins  --output-directory checkm2  -x fa --database_path ~/db/checkm2/CheckM2_database/uniref100.KO.1.dmnd
```

--input bins 输入数据为包含bins的文件夹
-x fa bins的扩展名文件格式为fa
--database_path 手动指定下载的数据库路径

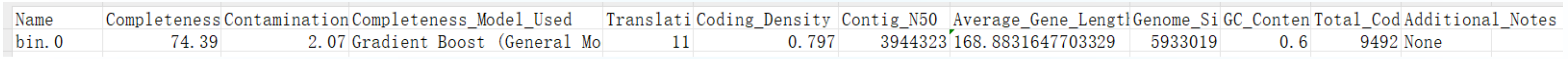
checkm2：结果展示

CheckM2输出文件夹：



```
checkm2.log   diamond_output   protein_files   quality_report.tsv
```

quality_report文件内容：



Name	Completeness	Contamination	Completeness_Model_Used	Translati	Coding_Density	Contig_N50	Average_Gene_Lengtl	Genome_Si	GC_Conten	Total_Cod	Additional_Notes
bin.0	74.39	2.07	Gradient Boost (General Mo	11	0.797	3944323	168.8831647703329	5933019	0.6	9492	None

完整度和污染度为两个重要指标。
中等质量：Completeness > 50%; Contamination < 10%
高质量：Completeness > 90%; Contamination < 5%

gtdb-tk：对宏基因组bins进行物种注释

- 使用conda进行软件安装

```
conda create -n gtdbtk-2.2.6 -c conda-forge -c bioconda gtdbtk=2.2.6
```

- 查看软件版本: v2.2.6

```
conda activate gtdbtk-2.2.6
gtdbtk --version
```

- 使用conda的package进行软件安装

```
cd ~/tools
wget -c --no-check-certificate --no-proxy
https://figshare.com/ndownloader/files/45672426 -O gtdbtk-2.2.6.tar.gz
mkdir ~/miniconda3/envs/gtdbtk-2.2.6/
tar -xzvf gtdbtk-2.2.6.tar.gz -C ~/miniconda3/envs/gtdbtk-2.2.6/
conda activate gtdbtk-2.2.6
conda unpack
```

gtdb-tk：数据库配置

- 配置软件数据库

```
mkdir -p ~/db/gtdbtk && cd ~/db/gtdbtk
wget -c
https://data.ace.uq.edu.au/public/gtdb/data/releases/latest/auxillary_files/gtdbtk_da
ta.tar.gz
tar -zxvf gtdbtk_data.tar.gz
```

- 设置数据库路径

```
export GTDBTK_DATA_PATH=~/db/gtdbtk/release214
```

gtdb-tk：使用案例

- 检查软件依赖

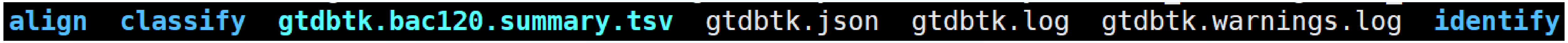
```
gtdbtk check_install
```

- 运行bin物种分类及进化树构建

```
gtdbtk classify_wf --genome_dir maxbin2_bins/ --extension fa  --skip_ani_screen --out_dir gtdbtk
gtdbtk convert_to_itol --input some_tree.tree --output itol.tree
```

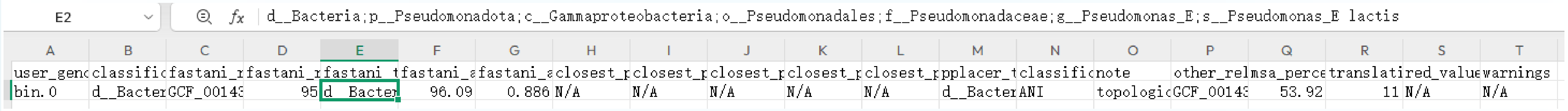
gtdb-tk：结果展示

gtdb-tk 输出文件夹：



```
align   classify   gtdbtk.bac120.summary.tsv   gtdbtk.json   gtdbtk.log   gtdbtk.warnings.log   identify
```

gtdb-tk 对bins的注释结果：



E2 　 *fx* 　 d__Bacteria;p__Pseudomonadota;c__Gammaproteobacteria;o__Pseudomonadales;f__Pseudomonadaceae;g__Pseudomonas_E;s__Pseudomonas_E lactis

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	user_genc	classifi	fastani_r	fastani_r	fastani_t	fastani_a	fastani_a	closest_p	closest_p	closest_p	closest_p	closest_p	pplacer_t	classific	note	other_rel	msa_perce	translati	red_value	warnings
2	bin.0	d__Bacter	GCF_00143	95	d__Bacter	96.09	0.886	N/A	N/A	N/A	N/A	N/A	d__Bacter	ANI	topologic	GCF_00143	53.92	11	N/A	N/A

总结

- EasyNanoMeta是专门为纳米孔宏基因组数据分析设计的软件，集成了全流程分析与逐步分析；
- 借助abricate与EasyNanoMeta中的脚本也可从reads水平对纳米孔宏基因组数据进行功能基因注释与定量；
- metaFlye为最佳的纳米孔宏基因组组装软件；
- SemiBin在纳米孔长读宏基因组分箱方面性能较为均衡；
- 纳米孔长读宏基因组测序数据与短读测序数据结合分析时，许多分析方法与二代短读宏基因组分析相通。

扫码关注生信宝典，学习更多生信知识

扫码关注宏基因组，获取专业学习资料

易生信，没有难学的生信知识