1. NCBI
   1. Literature

Bookshelf：NCBI Bookshelf 提供生命科学和卫生保健书籍、报告和数据库的全文，主要用于研究和教育。

资源分类

PubMed：NCBI最著名的数据库，包含数百万篇生物医学文献摘要和一些全文。

PubMed Central (PMC)：一个免费的数字存档，保存了期刊文献的全文。

MeSH：Medical Subject Headings，用于文献分类的医学主题词汇表。

NLM Catalog：包含美国国立医学图书馆（NLM）收藏的书籍、期刊和其他出版物的目录信息。

基因和基因组信息

* 1. Genes

Gene：基因数据库，提供基因的详细信息。

GEO DataSets：Gene Expression Omnibus 数据集，包含高通量基因表达数据。

GEO Profiles：基因表达轮廓，显示单个样本的表达模式。

PopSet：群体集，涉及多样本的基因序列集合。

Assembly / Genome：基因组组装和基因组信息。

BioCollections：生物集合，信息主要来自生物样品的整理和分类。

* 1. Protein

蛋白质信息

Proteins：与蛋白质相关的数据库，包括序列和结构信息。

Conserved Domains：保守结构域，蛋白质中具有保守功能的区域。

Identical Protein Groups：相同蛋白质群组，包含序列完全相同的蛋白质。

Conserved Domains：保守结构域数据库包含了蛋白质中具有保守功能的区域信息。保守结构域是指在进化过程中保持不变的蛋白质片段，这些片段通常与特定的生物功能相关。这个数据库用于识别和注释蛋白质序列中的功能域。

Identical Protein Groups：相同蛋白质群组数据库包含序列完全相同的蛋白质群组。它汇集了在不同物种或不同样本中发现的具有完全相同氨基酸序列的蛋白质。这个数据库有助于研究同一蛋白质在不同背景下的表达和功能。

Protein：蛋白质数据库提供单个蛋白质的详细信息，包括其序列、功能、来源物种、参考文献等。研究人员可以使用这个数据库来查找特定蛋白质的相关信息。

Protein Family Models：蛋白质家族模型数据库包含用于识别和分类蛋白质家族的模型和工具。蛋白质家族是由共同祖先蛋白演化而来的蛋白质组，通常具有相似的结构和功能。这个数据库帮助识别新的蛋白质家族成员并研究其功能。

Structure：结构数据库包含生物分子的三维结构信息，特别是蛋白质的三维结构。这些结构通常通过X射线晶体学、核磁共振（NMR）或电子显微镜（EM）等方法确定。了解蛋白质的结构对于理解其功能至关重要。

* 1. Genomes

Assembly / Genome：这部分包含基因组组装数据。基因组组装是将短的DNA序列片段组合成更长的连续序列，从而创建完整的基因组序列。它包括各种生物物种的参考基因组和最新组装版本。

BioCollections：这个数据库是生物样品的分类集合，涵盖了不同生物物种的样本信息。

BioProject：BioProject数据库包含与特定研究项目相关的信息，这些项目通常涉及基因组学、表观基因组学或其他高通量测序研究。

BioSample：BioSample数据库保存了生物样品的元数据，这些样品与基因组学、转录组学、蛋白质组学等研究相关。

Nucleotide：这个数据库包含核苷酸序列的数据，包括DNA和RNA序列。6,052表示有6,052条相关的核苷酸序列记录。

SRA：Sequence Read Archive 是一个大型数据库，保存了高通量测序（如二代测序）生成的原始读数数据。

Taxonomy：分类学数据库包含关于生物分类的信息，包括物种的科学名称及其分类层级。

* 1. Clinical

ClinicalTrials.gov：临床试验数据库。

ClinVar：临床基因变异数据库。

dbGaP：数据库，用于存储与基因型和表型相关的数据。

dbSNP：单核苷酸多态性（SNP）数据库。

dbVar：结构变异数据库。

GTR：遗传测试注册信息。

MedGen：医学遗传学数据库。

OMIM：在线人类孟德尔遗传数据库。

* 1. PubChem

BioAssays：生物测定，存储与化合物测试相关的数据。

Compounds：化合物数据库。

Pathways：生物通路数据库。

Substances：物质数据库，涵盖各种化学物质的信息。

1. Linux 解压命令

tar -xvf file.tar 解压 tar包

tar -xzvf file.tar.gz 解压tar.gz

tar -xjvf file.tar.bz2 解压 tar.bz2

tar -xZvf file.tar.Z 解压tar.Z

unrar e file.rar 解压rar

unzip file.zip 解压zip

tar -xzvf sratoolkit.3.1.1-ubuntu64.tar.gz

tar -xzvf k2\_standard\_20240904.tar.gz

1. 方法总结

使用 sratoolkit (v2.9.6) 中的 fastq-dump (v2.9.6) 和选项“—split-3”将所有下载的 SRA 文件拆分为双端原始读取，然后使用 Trimmomatic66 单独对所有原始读取进行质量控制(v2.39) 修剪接头和引物，并过滤短 (<50 bp) 和低质量读数 (<20 个碱基)，然后使用 MEGAHIT67 (v1.2.9) 进行组装，最小重叠群长度为 500 bp，分别对每个样本使用选项“--k-step 10 --k-min 27”。

1. 路径

主路径 ~/gemine

cd ~/gemine

1. sratoolkit手册

github:

<https://github.com/ncbi/sra-tools/wiki>

目前路径是

D:\hty\creat\paper\do\HJun\Tool\sratoolkit.3.1.1-win64\bin\prefetch.exe

测试用 ERR1309574

* 1. 下载sra

D:\hty\creat\paper\do\HJun\Tool\sratoolkit.3.1.1-win64\bin\prefetch.exe ERR1309574

/home/hty/ins/sratoolkit.3.1.1-ubuntu64/bin/prefetch SRR30410765 -O /mnt/f/SRA\_fasterq --max-size u

prefetch ERR1309574 -O /path/to/be/used

D:\hty\creat\paper\do\HJun\Tool\sratoolkit.3.1.1-win64\bin\prefetch.exe ERR1309574 -O C:\Users\99791\Desktop

/home/hty/ins/sratoolkit.3.1.1-ubuntu64/bin/prefetch ERR1309574 -O /mnt/c/Users/99791/Desktop

/home/hty/ins/sratoolkit.3.1.1-ubuntu64/bin/prefetch ERR1309574 -O /mnt/d

确保 /path/to/be/used 的最后一个目录是接入本身。例如，prefetch SRR000001 -O path/to/be/used/SRR000001

SRA 工具期望运行的所有文件SRR000001都存储在与接入同名的目录中：SRR000001。它被称为“接入作为目录”。

当前路径下的a文件夹：-O a

-toolprefetch的默认最大下载大小为20G。如果请求的访问量大于20G，则需要增加该限制。无论请求的访问量有多大，您都可以指定一个极高的限制。您还可以使用 -toolvdb-dump和--info选项查询访问量大小。例如，vdb-dump SRR000001 --info告诉您此访问量有多大（以及其他信息）。访问量SRR000001为932,308,473字节，低于默认限制，因此无需进一步操作。访问量SRR1951777为410,112,373,995字节。要下载此访问量，您必须将限制提升到该大小以上：

无限制：--max-size u

prefetch ERR1309574 --max-size u

* 1. 从 SRA - 接入中提取 fastq 文件

D:\hty\creat\paper\do\HJun\Tool\sratoolkit.3.1.1-win64\bin\fasterq-dump SRR30995594 --split-3

D:\hty\creat\paper\do\HJun\Tool\sratoolkit.3.1.1-win64\bin\fasterq-dump C:\Users\99791\Desktop\SRR22183615 --split-3 --outdir C:\Users\99791\Desktop

fasterq-dump ERR1309574 --split-3

双端测序 --split-3 : 将双端测序分为两份,放在不同的文件。对于一个你不知道到底是单端还是双端的SRA文件,一律用--split-3.

如果您希望在不同的目录中创建输出文件，请使用该--outdir选项。

当前路径下的a文件夹：--outdir a

总结

fasterq-dump ERR1309574 --split-3

fasterq-dump ERR1309574 --split-3 --outdir a

D:\hty\creat\paper\do\HJun\Tool\sratoolkit.3.1.1-win64\bin\fasterq-dump "SRR23005778" --split-3

D:\hty\creat\paper\do\HJun\Tool\sratoolkit.3.1.1-win64\bin\fasterq-dump "ERR1309574" --split-3

1. fastp

<https://github.com/OpenGene/fastp>

安装

conda install -c bioconda fastp

fastp支持单端（SE）和双端（PE）输入/输出。

对于 SE 数据，您只需通过-i或指定读取1 输入，并通过或-i/--in1指定读取1 输出。-o/--out1

对于 PE 数据，还应该通过-I或指定 read2 输入，并通过或-I/--in2 输出。-O/--out2

如果您不指定输出文件名，则不会写入任何输出文件，但仍会对过滤前后的数据进行 QC。

如果文件名以以下形式结尾，则输出将被 gzip 压缩.gz

单端

fastp -i in.fq -o out.fq

双端

fastp --in1 /mnt/c/Users/99791/Desktop/SRR13503329/SRR13503329\_1.fastq –in2 /mnt/c/Users/99791/Desktop/SRR13503329/SRR13503329\_2.fastq --out1 out.fq

fastp \

-i /mnt/c/Users/99791/Desktop/SRR13503329/SRR13503329\_1.fastq \

-o /mnt/c/Users/99791/Desktop/SRR13503329/out1.fastq \

-I /mnt/c/Users/99791/Desktop/SRR13503329/SRR13503329\_2.fastq \

-O /mnt/c/Users/99791/Desktop/SRR13503329/out2.fastq \

--thread 20 \

--cut\_front --cut\_tail \

--length\_required 50 \

--qualified\_quality\_phred 20 \

--unqualified\_percent\_limit 40 \

--html /mnt/c/Users/99791/Desktop/SRR13503329/report.html --json report.json

-i input.fastq：输入文件（可以是单端或双端数据）。

-o output.fastq：输出文件。

--cut\_front 和 --cut\_tail：去除 reads 的前端和/或尾部接头。

--length\_required 50：只保留长度大于或等于 50 bp 的 reads。

--qualified\_quality\_phred 20：只保留质量值（Phred score）大于等于 20 的碱基。

--unqualified\_percent\_limit 40：丢弃含有超过 40% 低质量碱基（质量值 <20）的 reads。

--html report.html：生成 HTML 格式的质量检查报告。

--json report.json：生成 JSON 格式的报告，方便后续分析。

fastp -i "/mnt/d/hty/creat/paper/do/HJun/Database/fastq/SRR30995602/SRR30995602\_1.fastq" -I "/mnt/d/hty/creat/paper/do/HJun/Database/fastq/SRR30995602/SRR30995602\_2.fastq" -o "/mnt/d/hty/creat/paper/do/HJun/Database/fastq\_better/SRR30995602/SRR30995602\_1.fastq" -O "/mnt/d/hty/creat/paper/do/HJun/Database/fastq\_better/SRR30995602/SRR30995602\_2.fastq" --thread 20

1. megahit

https://github.com/voutcn/megahit

安装

conda install -c bioconda megahit

# 双端序列组装：

megahit -1 pe\_1.fq -2 pe\_2.fq -o out

#-1：pair-end 1序列，-2 pair-end 2序列，-o输出目录

# 单端序列：

megahit -r single\_end.fq -o out

# 交错的双端序列：

megahit --12 interleaved.fq -o out

megahit -1 /mnt/c/Users/99791/Desktop/SRR30995595/SRR30995595\_1.fastq \

-2 /mnt/c/Users/99791/Desktop/SRR30995595/SRR30995595\_2.fastq -o /mnt/c/Users/99791/Desktop/a

其他常用参数

1）-m/–memory：构建SdBG可以使用的最大内存，可设置0-1，也即占总内存的分数，默认为0.9

2）-t/–num-cpu-threads：程序运行使用的核数

3）–out-prefix：输出结果文件的前缀，例如contig文件会是OUT\_DIR/OUT\_PREFIX.contigs.fa

4）–min-contig-len输出的最短contigs，默认为200

5）–test 在自带测试数据上执行组装，常用于测试软件安装是否正常

final.contigs.fa: 组装结果，fasta格式

log: megahit程序运行时的log日志，便于查看具体执行进度和排错

options.json: 执行组装程序参数的json格式

intermediate\_contigs: 中间组装结果

megahit -1 "/mnt/d/hty/creat/paper/do/HJun/Database/fastq\_better/SRR30995602/SRR30995602\_1.fastq" -2 "/mnt/d/hty/creat/paper/do/HJun/Database/fastq\_better/SRR30995602/SRR30995602\_2.fastq" -o /mnt/d/hty/creat/paper/do/HJun/Database/megahit/SRR30995602 -t 20

megahit -1 sample\_1\_R1.fastq -2 sample\_1\_R2.fastq \

-o output\_directory \

--k-step 10 \

--k-min 27 \

--min-contig-len 500

--k-step 10：设置 k-mer 递增的步长为 10。

--k-min 27：设置最小 k-mer 为 27。

--min-contig-len 500：设置最小 contig 长度为 500 bp，短于此长度的 contig 会被过滤掉。

1. kraken2

https://github.com/DerrickWood/kraken2

<https://blog.csdn.net/m0_53945548/article/details/133800418>

数据库位置

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

务必下载最新的

conda remove kraken2

conda update conda

conda update --all

conda config --add channels conda-forge

conda install bioconda::kraken2=2.1.3

kraken2 --version

--use-ftp

kraken2-build --download-library bacteria --db /mnt/f/Hjun/kraken\_db --threads 20

https://benlangmead.github.io/aws-indexes/k2 #数据库位置

--db kraken\_db：指定数据库存储目录，kraken\_db 目录将存储下载的细菌数据库。如果没有指定，Kraken2 会在当前目录下创建一个默认目录。

运行 kraken2

kraken2 --threads 20 --quick --paired \

--db /mnt/d/hty/down/db \

--report A1.kreport \

--output A1.kraken \

--memory-mapping \

/mnt/c/Users/99791/Desktop/SRR22183625\_1.fastq \

/mnt/c/Users/99791/Desktop/SRR22183625\_2.fastq

kraken2 /mnt/d/hty/creat/paper/do/HJun/Database/fastq\_better/SRR29303764/SRR29303764.fastq --quick --db /mnt/d/hty/creat/paper/do/HJun/Tool/kraken2\_db --threads 20 --report /mnt/d/hty/creat/paper/do/HJun/Database/kraken2/SRR29303764/kraken2.report --output /mnt/d/hty/creat/paper/do/HJun/Database/kraken2/SRR29303764/kraken2.output --memory-mapping /mnt/d/hty/creat/paper/do/HJun/Database/fastq\_better/SRR29303764/SRR29303764.fastq

会生成一个output文件和一个report文件，接下来小果为大家一一解读一下。 其中output文件总共有五列（如下图）。第一列：C/U，代表是否分类，C代表分类，U则代表未分类。第二列：测序数据read的 ID号 ，来自FASTA/FASTQ文件。第三列：分类ID号，（例：Mucilaginibacter ginsenosidivorax (分类id为862126);如果分类id号为0,说明没有分类）。第四列：测序read的长度。第五列：以空格分隔的列表，指示序列中每个k-mer的LCA映射 （冒号之前taxid，0为没有数据）

为了解决这个问题，请将此标志与其他标志一起使用 --memory-mapping

因此，你有类似这样的内容：

kraken2 --use-names --memory-mapping --db /mnt/data/kraken\_db/ --threads 8 --report kraken.report --paired R1.fastq R2.fastq > kraken.output

它的作用是避免将数据库加载到 RAM 中，从而减少加载数据库所需的内存占用。

1. Bracken

conda install -c bioconda bracken

用git构建时

Permission denied 错误通常是由于没有执行权限造成的。你可以通过以下步骤解决问题：

步骤 1: 检查文件权限

确认 bracken 文件是否具有可执行权限：

ls -l /home/hty/ins/Bracken/bracken

如果没有执行权限（x），你会看到权限类似 -rw-r--r--，说明文件不能直接执行。

步骤 2: 添加执行权限

为 bracken 文件添加可执行权限：

chmod +x /home/hty/ins/Bracken/bracken

步骤 3: 尝试再次运行

重新尝试运行 Bracken：

/home/hty/ins/Bracken/bracken -h

Bracken参数解释

-d，Kraken2数据库路径（包含Braken对应长度索引）；

-i，Kraken2的输出文件名（--report的输出文件名），在这里作为输入文件；

-o，Bracken输出文件（校正详情）文件名；

-w，Bracken计算后的新报告（每个物种的reads数目）文件名；

-r， reads长度；

-l，分类水平（D,P,C,O,F,G,S）

bracken \

-d /mnt/d/hty/down/db \

-i A1.kreport \

-o A1.bracken.S \

-l G \

K（界级）、P（门）、C（纲）、O（目）、F（科）和 G（属）。

bracken -d /mnt/d/hty/creat/paper/do/HJun/Tool/kraken2\_db -i /mnt/d/hty/creat/paper/do/HJun/Database/kraken2/SRR29303764/kraken2\_report.txt -o /mnt/d/hty/creat/paper/do/HJun/Database/bracken/SRR29303764/bracken\_report.txt -l G

1. fastqc

质量评估

conda install -c bioconda fastqc

fastqc SRR29303764.fastq

https://blog.csdn.net/qq\_74093550/article/details/131530092

安静输出

fastqc SRR2932715.fastq --quiet

fastqc \*.fastq

-o 用来指定输出文件的所在目录，生成的报告的文件名是根据输入来定的，注意是不能自动新建目录的。输出的结果是.zip文件，默认不解压缩，命令里加上--noextract则解压缩。

-f 用来强制指定输入文件格式，默认自动检测。支持fastq、bam、sam极相应的gz压缩格式

-c 污染物选项，输入的是一个文件，格式是Name[Tab] Sequence，#开头的行是注释，里面是可能的污染序列，如果有这个选项，FastQC会在计算时候评估污染的情况，并在统计的时候进行分析。

-q 会进入沉默模式，指定这个选项的时候，程序不会实时报告运行的状况，即不出现下面的提示

1. SemiBin

conda install -c conda-forge -c bioconda semibin

# 辅助semibin的软件下载

conda install -c bioconda bowtie2

conda install -c bioconda samtools

conda install -c bioconda megahit

1. metabat2

bowtie2-build -f final.contigs.fa contigs

bowtie2 -q -x contigs -1 SRR22183625\_1.fastq -2 SRR22183625\_2.fastq -S alignment.sam -p 20

bowtie2-build final.contigs.fa contigs

bowtie2 -x contigs -1 SRR22183625\_1.fastq -2 SRR22183625\_2.fastq -S alignment.sam -p 20

samtools view -bS alignment.sam > alignment.bam

samtools sort alignment.bam -o sorted\_alignment.bam

samtools index sorted\_alignment.bam

jgi\_summarize\_bam\_contig\_depths --outputDepth depth.txt sorted\_alignment.bam

metabat2 -i final.contigs.fa -a depth.txt -o bins\_dir/bin

1. Diamond

conda install -c bioconda diamond=2.1.8

diamond --version

常用功能基因数据库包括：

KEGG：对代谢途径的注释。

COG 或 eggNOG：注释进化保守的基因功能。

Pfam：以蛋白域为基础的功能注释。

CARD/ResFinder：药物抗性基因注释。

VFDB：毒力基因注释。

搜索。就两个子命令，blastp和blastx，前者比对蛋白，后者比对DNA序列

核酸序列建议用 DIAMOND blastx。

如果已经组装得到蛋白序列，用 DIAMOND blastp

-d swissprot：使用刚刚构建的 Swiss-Prot 数据库。

-q test.fasta：输入查询序列。

-o results.txt：输出比对结果文件。

-f 6：输出为标准化 BLAST 表格式，默认就好

-e：设置比对结果显著性阈值，默认就好

diamond blastx -d swissprot -q test.fasta -o results.txt -f 6 -e 1e-5 --threads 4

/mnt/c/Users/99791/Desktop/SRR15526926\_1.fastq

diamond makedb --in /mnt/d/hty/down/gene.fna -d swissprot

diamond blastx -d /home/hty/ins\_bao/swissprot.dmnd -q /mnt/c/Users/99791/Desktop/final.contigs.fa -o results\_1.txt --threads 20

diamond blastx -d /mnt/d/hty/down/gene.fna -q /mnt/c/Users/99791/Desktop/final.contigs.fa -o results\_1.txt --threads 20

结果说明

字段说明：

Query: 查询序列名称。

Subject: 匹配到的蛋白序列编号（如 Swiss-Prot 中的 ID）。

相同匹配的百分比

对齐长度

其他字段：具体比对位置信息等。

wget <https://ftp.ncbi.nlm.nih.gov/blast/db/FASTA/nt.gz>

1. BLAST

conda install -c bioconda blast

blastn -version

* 1. megaBlast

对于linux本地版本：

很简单，只要在你原来的blastn脚本里面加上一个参数：-task megablast

* 1. 构建库

makeblastdb -in sequences.fasta -dbtype nucl -out pheA\_db

makeblastdb -in /mnt/d/hty/creat/paper/do/HJun/Database/blast\_db\_nucl.fasta -dbtype nucl -out /mnt/d/hty/creat/paper/do/HJun/Tool/blast\_db

-in sequences.fasta：指定输入的 FASTA 文件。

-dbtype nucl：指定数据库类型，nucl 代表核酸序列（如果是蛋白质序列，使用 prot）。

-out my\_blast\_db：指定输出的数据库名称，这会生成数据库文件 my\_blast\_db，并在同一目录下创建相关的数据库文件。

* 1. 对比

核酸序列比对核酸数据库（blastn），核酸序列比对蛋白数据库（blastx），蛋白序列比对蛋白数据库（blastp）

/mnt/c/Users/99791/Desktop/ERR12711808\_1.fastq

blastn -query query.fasta -db pheA\_db -out blast\_result.txt -num\_threads 20 -outfmt 6 -task megablast -evalue 1e-5

-query： 输入文件路径及文件名

-out：输出文件路径及文件名

-db：格式化了的数据库路径及数据库名

-outfmt：输出文件格式，总共有12种格式，6是tabular格式对应之前BLAST的m8格式

-evalue：设置输出结果的e-value值

-num\_threads：线程数

核酸序列比对核酸数据库

* 1. nt 数据库

wget -p /mnt/c/Users/99791/Desktop <https://ftp.ncbi.nlm.nih.gov/blast/db/FASTA/nt.gz>

1. Pear

conda install pear

pear -f file\_R1.fq -r file\_R2.fq -o merged\_output

pear -f /mnt/d/hty/creat/paper/do/HJun/Database/fastq\_better/SRR28741039/SRR28741039\_1.fastq -r /mnt/d/hty/creat/paper/do/HJun/Database/fastq\_better/SRR28741039/SRR28741039\_2.fastq -o /mnt/d/hty/creat/paper/do/HJun/Database/pear\_fastq\_better/SRR28741039 -j 20

1. seqtk

conda install -c bioconda seqtk

seqtk seq -a reads.fastq > reads.fasta