

LEB 课程内容总结 沈南奇 2300934024

EMBOSS

EMBOSS 软件安装

EMBOSS 软件包主网站

<http://emboss.open-bio.org/>

进入网站没有直接找到下载方式

同组廖一岩分享了安装包，在运行该 exe 文件前提示需要先安装 java

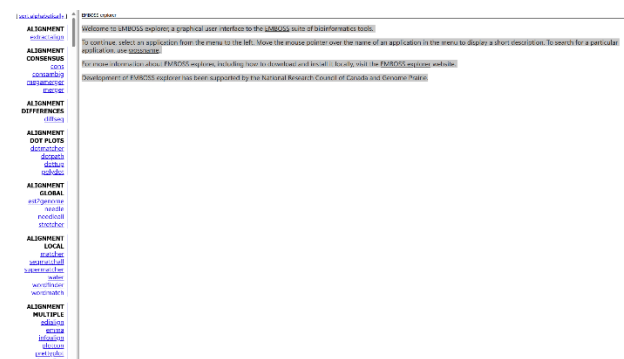
<https://www.java.com/zh-CN/download/manual.jsp>

另一种 EMBOSS 使用方式

荷兰瓦格宁根大学 EMBOSS Explorer

<http://www.bioinformatics.nl/emboss-explorer/>

通过浏览器访问该网站，打开后页面右侧可以选择工具打开使用



运行 needle 程序

交互式

```
(base) snq23@bbt:~/MyEMBOSS/input$ ls
HBA_HUMAN.FASTA  HBA_MOUSE.FASTA  HBA_RAT.FASTA  HBB_HUMAN.FASTA
(base) snq23@bbt:~/MyEMBOSS/input$ needle
Needleman-Wunsch global alignment of two sequences
Input sequence: HBA_HUMAN.FASTA
Second sequence(s): HBB_HUMAN.FASTA
Gap opening penalty [10.0]:
Gap extension penalty [0.5]:
Output alignment [hba_human.needle]:
(base) snq23@bbt:~/MyEMBOSS/input$ ls
HBA_HUMAN.FASTA  hba_human.needle  HBA_MOUSE.FASTA  HBA_RAT.FASTA  HBB_HUMAN.FASTA
```

操作流程：

1. 准备需要比对的两个序列，以人血红蛋白 HBA 和 HBB 亚基的氨基酸序列为例。
2. 输入 needle 开始运行程序。
3. 输入第一个研究序列 FASTA 格式文件的文件名。
4. 输入第二个文件名。
5. 系统给出起始空位罚分值 Gap opening penalty，默认 10.0，可输入自定义分数后回车，若直接回车则取用默认值。
6. 系统给出延伸空位罚分值 Gap extension penalty，默认 0.5，输入自定义分数后回车，若直接回车则取用默认值。
7. 系统提示输出文件名，默认小写，回车确定，也可自行输入。
8. 当前目录下多出 hba_human.needle 文件，可 cat 命令查看

```
(base) snq23@bbt:~/MyEMBOSS/input$ cat hba_human.needle
#####
# Program: needle
# Rundate: Wed 17 Apr 2024 00:12:24
# Commandline: needle
#   -asequence HBA_HUMAN.FASTA
#   -bsequence HBB_HUMAN.FASTA
# Align_format: srspair
# Report_file: hba_human.needle
#####
#
# Aligned_sequences: 2
# 1: HBA_HUMAN
# 2: HBB_HUMAN
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 149
# Identity:      65/149 (43.6%)
# Similarity:   90/149 (60.4%)
# Gaps:         9/149 ( 6.0%)
# Score: 292.5
#
#
#####
HBA_HUMAN      1  MV-LSPADKTNVKAAGKVGAGHAGEYGAELERMFSLFPTTKTYFPHF-D  48
  || |:|:|:|:|:| ||| :..|:|:|:|:|:|:|:|:|:|:|:|
HBB_HUMAN      1  MVHLTPEEKSAVTALWGKV--NVDEVGGEALGRLLVVYPWTQRFFESFGD  48

HBA_HUMAN     49  LS-----HGSAQVKHGKGVADALTNAAHVDDMPNALSADLHAHKLR  93
  || .|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|
HBB_HUMAN     49  LSTPDVAVGNPKVKAHGKKVLGAFSDGLAHLONLKGTFATLSELHCDKLH  98

HBA_HUMAN     94  VDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR  142
  |||:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|:|
HBB_HUMAN     99  VDPENFRLLGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVANALAHKYH  147

#-----
#
```

```
(base) snq23@bbt:~/MyEMBOSS/input$ needle HBA_HUMAN.FASTA HBB_HUMAN.FASTA -gapo 10.0 -gape 0.5 -out HBA_HUMAN_HBB.needle
Needleman-Wunsch global alignment of two sequences
```

参数式

操作：

输入 needle HBA_HUMAN.FASTA HBB_HUMAN.FASTA -gapo 10.0 -gape 0.5 -out HBA_HUMAN_HBB.needle

备注：

红字部分为要比对的序列文件，-gapo 10.0 是起始空位罚分，-gape 0.5 是延伸空位罚分，-out HBA_HUMAN_HBB.needle 是输出名为 HBA_HUMAN_HBB.needle 的文件（下图第二个）

```
(base) snq23@bbt:~/MyEMBOSS/input$ ls
HBA_HUMAN.FASTA  HBA_HUMAN_HBB.needle  hba_human.needle  H
```

运行 dottup 点阵图程序

点阵图程序

程序	功能
Dottup	用点阵图方式显示两条序列间相似性区域或同一序列中的重复片段，相似性度量基于相同位点；主要参数为字长Word size，默认值10，多用于核酸序列。
Dotpath	用点阵图方式显示两条序列间相似性区域，相似性度量基于相同位点，只显示主对角线相似性区域，不显示重复性短片段；主要参数为字长Word size，默认值4，可用于核酸或蛋白质序列。
Dotmatcher	用点阵图方式显示两条蛋白质序列间相似性区域或同一序列中重复片段，相似性度量基于计分矩阵，默认为BLOSUM62；主要参数：1) 滑动窗口大小Window size，默认值10，2) 相似性阈值threshold，默认值23。
Polydot	用点阵图方式显示多条序列之间的相似性区域，相似性度量基于相同位点；主要参数为字长Word size，默认值6；可用于核酸或蛋白质序列

知乎搜到资料：

dottup：一个 word 要在两个序列里面完全一模一样才有 dot

dotmatcher：不完全匹配，只要在 threshold 以上，就有 dot

dotpath：类似 dottup (但是默认的 word size 比较小)，但是序列里面的任意一段只能 match 到一个地方 (不能对到两个不同的地方)。换句话说，没有 overlap 的 alignment，也就是从图上看，任意 x 值上都只有一个 dot，任意 y 值上也只有一个 dot。

polydot：一个 fasta 文件，里面所有序列对 (all pairwise) 的 dottup。比如有 3 条序列就有 3*3 个 dottup

作者：Philip Yang

链接：<https://www.zhihu.com/question/28623438/answer/41580497>

点阵图程序 dottup 多用于核酸序列，而 dotmatcher 则可用于蛋白质序列 (EMBOSS 软件包序列分析程序应用实例，罗静初，DOI: 10.12113/202008002，生物信息学)

交互式

```
(base) snq23@bbt:~/MyEMBOSS/input$ dottup
Display a wordmatch dotplot of two sequences
Input sequence: HBA_HUMAN.FASTA
Second sequence: HBB_HUMAN.FASTA
Word size [10]:
Graph type [x11]: svg
Created dottup.svg
```

操作流程：

- 1.准备 HBA_HUMAN.FASTA 和 HBB_HUMAN.FASTA 文件
- 2.输入 dottup 运行程序

- 3.输入第一步准备文件的文件名
- 4.修改字长或选取默认值 10
- 5.输出结果图形格式默认 X11, 输入 svg ([SVG: 可缩放矢量图形](#))
- 6.生成 dottup.svg 文件

参数式

```
(base) snq23@bbt:~/MyEMBOSS/input$ dottup HBA_HUMAN.FASTA HBB_HUMAN.FASTA
-word 3 -graph svg -goutfile HBA-HBB
Display a wordmatch dotplot of two sequences
Created HBA-HBB.svg
```

操作: 输入 dottup HBA_HUMAN.FASTA HBB_HUMAN.FASTA -word 3 -graph svg -goutfile HBA-HBB

备注: 输入 dottup, 两个文件名, 设置字长 (-word) 和输出结果图形格式 (-graph), 修改输出结果文件名 (-goutfile)。默认输出图形格式是 x11, 但是默认格式生成不成功, 需修改成 svg 格式

```
(base) snq23@bbt:~/MyEMBOSS/input$ dottup
Display a wordmatch dotplot of two sequences
Input sequence: HBA_HUMAN.FASTA
Second sequence: HBB_HUMAN.FASTA
Word size [10]:
Graph type [x11]:
Can't open display
```

GitHub

GitHub 重要性

软件开发的核心工具

项目管理的优选平台

开源协作的中心舞台

GitHub 是一个存放软件代码的网站, 将代码放在网站上, 其他人可以在现有代码的基础上进行二次开发, 不用从头开始工作。如果是团队合作, 团队成员可以上传自己的工作, 也可以拉取他人的。

基本概念

仓库 & 分支:

仓库(Repository): 项目的存储地点

The screenshot shows a GitHub profile for Linlin Yan (颜林林). The 'Repositories' tab is selected and highlighted with a yellow box. The profile includes a bio, location (Beijing, China), and a list of repositories. The repositories listed are:

- seqpipe**: A framework for SEQuencing data analysis PIPELines. (Public, 157 repositories, 2.3k stars, updated 2 weeks ago)
- bukaopuyanlun**: 个人微信公众号“不靠道颜论”的文章链接目录. (Public, 1 star, updated 3 weeks ago)
- simple-style**: My simple style hugo theme, based on <https://yanlinlin82.github.io/webpage-templates/simple-style/index.html>. (Public, 30 stars, 13 forks, updated last month)
- share**: (Public, updated last month)
- simple-style-demo**: (Public, updated on Mar 20)

分支(Branch): 不同版本的开发线路

主分支: 主开发线

特性分支: 新功能开发

Branches

Overview Active Stale All					
Q Search branches...					
Default					
Branch	Updated	Check status	Behind Ahead	Pull request	
master	5 years ago			Default	
Active branches					
Branch	Updated	Check status	Behind Ahead	Pull request	
dev-6	2 weeks ago		0 53		

提交 & 拉取请求:

提交(Commit):

保存更改的快照

提交历史: 更改的记录

拉取请求(Pull Request):

合并分支的请求

代码审查: 团队成员评审更改

合并: 将更改并入主分支


主页 overview

Product Solutions Open Source Pricing

Search or jump to...

Sign in Sign up

Overview Repositories 157 Projects Packages Stars 2.3k



Linlin Yan (颜林林)

yanlinlin82

Follow

Bioinfo PhD, with 20+ yrs of programming experience, also an open-source advocate, interested in NGS, genomics and other -omics. Wechat: yanlinlin82. 公众号: 不靠道颜论

347 followers 586 following

Bioinformatics Assistance Tech Co Ltd

Beijing, China

17:21 - same time

<https://yanlinlin.cn/>

@yanlinlin82

in:yanlinlin82

yanlinlin82

<https://www.kaggle.com/linlinyan>

yanlinlin82 / README.md

- Hi, I'm @yanlinlin82
- I'm interested in ...
- I'm currently learning ...
- I'm looking to collaborate on ...
- How to reach me ...

Pinned

gvenn

Public

Venn Diagram by ggplot2, with really easy-to-use API.

R

153

22

EMBOSS

Public

WARNING: This is NOT OFFICIAL source repository of EBI EMBOSS (The European Molecular Biology Open Software Suite)! I use this repository to record some modifications/bug fixings.

C

2

simple-style

Public

My simple style hugo theme, based on <https://yanlinlin82.github.io/webpage-templates/simple-style/index.html>

HTML

30

13

seqpipe

Public

A framework for SEQuencing data analysis PIPELines.

Perl

7

4

XCAVATOR

Public

A CNV caller of NGS data. This repo is for recording bug-fixings. Original source code is from <https://sourceforge.net/projects/xcavator/>, which has no any modification since 2017-10-12.

R

3

share

Public

321 contributions in the last year

2024

321 contributions in the last year

Learn how we count contributions

Less More

上图展示在 github 的贡献和活跃度

Git

参考资料

[Git - Book \(git-scm.com\)](#)
[Learn Git Branching](#) 小游戏

配置 Git

服务器中有 git，省略安装步骤，安装可以参照 [Git - 安装 Git \(git-scm.com\)](#)

确认安装完成:

```
git
## usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
##           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-pat
h]
##           [-p | --paginate | --no-pager] [--no-replace-objects] [--bar
e]
##           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
##           <command> [<args>]
##
## These are common Git commands used in various situations:
##
## start a working area (see also: git help tutorial)
##   clone      Clone a repository into a new directory
##   init       Create an empty Git repository or reinitialize an existing
one
##
## work on the current change (see also: git help everyday)
##   add       Add file contents to the index
##   mv        Move or rename a file, a directory, or a symlink
##   reset     Reset current HEAD to the specified state
##   rm        Remove files from the working tree and from the index
##
## examine the history and state (see also: git help revisions)
##   bisect    Use binary search to find the commit that introduced a bug
##   grep      Print lines matching a pattern
##   log       Show commit logs
##   show      Show various types of objects
##   status    Show the working tree status
##
## grow, mark and tweak your common history
##   branch    List, create, or delete branches
##   checkout  Switch branches or restore working tree files
##
##   commit    Record changes to the repository
##   diff      Show changes between commits, commit and working tree, etc
##   merge     Join two or more development histories together
##   rebase    Reapply commits on top of another base tip
##   tag       Create, list, delete or verify a tag object signed with GP
G
##
## collaborate (see also: git help workflows)
##   fetch     Download objects and refs from another repository
##   pull      Fetch from and integrate with another repository or a loca
l branch
##   push      Update remote refs along with associated objects
##
## 'git help -a' and 'git help -g' list available subcommands and some
## concept guides. See 'git help <command>' or 'git help <concept>'
## to read about a specific subcommand or concept.
```

配置用户邮箱和用户名

```
git config --global user.email "your@email.com"
git config --global user.name "Your Name"
```

下载并设置一些 Git 的辅助插件

```
mkdir git_mod
cd git_mod

wget https://raw.githubusercontent.com/git/git/master/contrib/completion/git-completion.bash
wget https://raw.githubusercontent.com/git/git/master/contrib/completion/git-prompt.sh
```

随后，用 vim 打开 ~/.bashrc，进行一些设置：

1. 将 `\[\033[1;36m\]${__git_ps1 " (%s)}"` 插入到 PS1 环境变量中的 `\w` 后面。

```
if [ "$color_prompt" = yes ]; then
    PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[1;36m\]${__git_ps1 " (%s)}"\[\033[00m\]\$ '
else
    PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\[\033[1;36m\]${__git_ps1 " (%s)}"\$ '
fi
unset color_prompt force_color_prompt
```

一共添加在三处

第一次进行完修改.bashrc 的操作后，无法在进行任何操作，退出之后出现登录不上服务器的问题，助教认为可能是有操作失误影响了文件中的其他部分，用未修改的文件替换我家目录下文件后，可以正常登陆以及使用。所以修改.bashrc 的操作还是比较“危险”的，之后助教又提供了官方教程的方法供学习。

<https://git-scm.com/book/zh/v2/%E9%99%84%E5%BD%95-A%E5%9C%A8%E5%85%B6%E5%AE%83%E7%8E%AF%E5%A2%83%E4%B8%AD%E4%BD%BF%E7%94%A8-Git-Bash-%E4%B8%AD%E7%9A%84-Git>

2. 添加如下内容：

```
# setting git mod
. ~/git_mod/git-completion.bash
. ~/git_mod/git-prompt.sh
export GIT_PS1_SHOWDIRTYSTATE=1
```

保存并退出后，运行 `./bashrc` 激活这些设置内容。

建立一个使用 Git 进行版本控制的项目

初始化

首先，新建一个名为 `lyl18_play_git` 的目录（名字大家随意就好），并在其中新建一个包含一些内容的文件。

```
mkdir lyl18_play_git
cd lyl18_play_git

echo "some text" > file1.txt
cat file1.txt
## some text
```

2.1 git init

现在我们在这个目录中初始化 Git。

```
git init
## Initialized empty Git repository in /rd1/home/LEB2024/lyl18/lyl18_play_g
it/.git/
```

现在这里多了一个叫 `.git` 的隐藏目录，表明现在的工作目录已经在 Git 的控制之下：

```
## lyl18@bbt:~/lyl18_play_git (master #) $ ll
## total 16
## drwxr-xr-x 3 lyl18 leb 4096 Mar 25 00:52 ./
## drwxr-xr-x 8 lyl18 leb 4096 Mar 25 00:48 ../
## -rw-r--r-- 1 lyl18 leb  10 Mar 25 00:51 file1.txt
## drwxr-xr-x 7 lyl18 leb 4096 Mar 25 00:52 .git/
```

- *: 表示工作目录里追踪文件中有未暂存的内容。
- +: 表示工作目录里有已暂存但还未提交的内容。

git status

使用 `git status` 命令可以查看当前工作目录的状态，可以看到被识别出的文件是 `untracked` 的状态，是红字，表示还没有被 Git 所追踪。并且提示我们可以使用 `git add` 对其进行追踪，以进行提交。

git status

对这个文件运行 `git add` 命令，Git 会把它放到一个“暂存区”，`git status` 检查，变成绿字了

git commit

这个命令表示将所有被暂存的改动储存起来，变成一次提交给记录下来。

```
git commit -m 'add file1'
## [master (root-commit) f348e18] add file1
## 1 file changed, 1 insertion(+)
## create mode 100644 file1.txt

git status
## On branch master
## nothing to commit, working tree clean
```

git log

通过 git log 命令查看提交的记录

我们现在已经进行了第一次提交，可以通过 git log 命令看一下提交的记录：

```
git log

## commit f348e188ddfd9fd6b8362c58a024e26849ff7aed (HEAD -> master)
## Author: LYL <lyuyulin@qq.com>
## Date:   Mon Mar 25 01:17:56 2024 +0800
##
##      add file1
```

这里可以看到 log 中保存了这次提交的各种信息，包括：

- **f348e188ddfd9fd6b8362c58a024e26849ff7aed**：一个哈希值来作为这次提交的“名字”，这个名字一般只需要前几位就可以**唯一地**表示这次提交（当然也有极小的可能出现重复）。*在一些撤销相关的操作里面需要输入指定的提交时，只需要这个名字的前几位就可以了。*
- **(HEAD -> master)**：**master** 在哪个提交的右边表示 **master** 分支最后一次提交是哪次，**HEAD** 指向 **master** 表示当前的工作环境是基于 **master** 分支的最后一次提交。一般不用管这些信息，只有当存在多个分支产生了不兼容的提交（即对同一个文件的同一块内容进行了不同的改动），或者本地和远程的分支不同步（产生了互相不兼容的提交）的时候，才需要观察这些信息。
- **Author: LYL <lyuyulin@qq.com>**：作者的信息（是我们一开始自己设置的）。
- **Date: Mon Mar 25 01:17:56 2024 +0800**：提交时间。
- **add file1**：git commit 时输入的信息。

如想撤销修改，可以回溯至之前的提交

Anaconda

使用 Conda

安装

服务器上有下载好的 Conda 存放于：/rd1/home/public/RNA-Seq/software/Miniconda3-latest-Linux-x86_64.sh

```
cp /rd1/home/public/RNA-Seq/software/Miniconda3-latest-Linux-x86_64.sh ~/
```

```
cd
```

```
bash Miniconda3-latest-Linux-x86_64.sh
```

一直按回车键，直到出现：

```
Do you accept the license terms? [yes|no]
[no] >>>
```

输入 yes 再回车

重新登陆服务器后左下角会出现 base 则表示安装完成

conda --help

```
(pro) snq23@bbt:~$ conda --help
usage: conda [-h] [-V] command ...

conda is a tool for managing and deploying applications, environments and packages.

Options:

positional arguments:
  command
  clean                Remove unused packages and caches.
  config              Modify configuration values in .condarc. This is modeled
                    after the git config command. Writes to the user .condarc
                    file (/rd1/home/LEB2024/snq23/.condarc) by default.
  create             Create a new conda environment from a list of specified
                    packages.
  help               Displays a list of available conda commands and their help
                    strings.
  info              Display information about current conda install.
  init              Initialize conda for shell interaction. [Experimental]
  install           Installs a list of packages into a specified conda
                    environment.
  list              List linked packages in a conda environment.
  package           Low-level conda package utility. (EXPERIMENTAL)
  remove            Remove a list of packages from a specified conda environment.
  uninstall         Alias for conda remove.
  run              Run an executable in a conda environment. [Experimental]
  search            Search for packages and display associated information. The
                    input is a MatchSpec, a query language for conda packages.
                    See examples below.
  update            Updates conda packages to the latest compatible version.
  upgrade           Alias for conda update.

optional arguments:
  -h, --help        Show this help message and exit.
  -V, --version     Show the conda version number and exit.

conda commands available from other packages:
  env
```

Conda 镜像站

[anaconda](#) | [镜像站使用帮助](#) | [清华大学开源软件镜像站](#) | [Tsinghua Open Source Mirror](#)

Anaconda 镜像使用帮助

Anaconda 是一个用于科学计算的 Python 发行版，支持 Linux, Mac, Windows, 包含了众多流行的科学计算、数据分析的 Python 包。

Anaconda 安装包可以到 <https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/> 下载。

TUNA 还提供了 Anaconda 仓库与第三方源（conda-forge、msys2、pytorch等，[查看完整列表](#)，更多第三方源可以前往[校园网联合镜像站查看](#)）的镜像，各系统都可以通过修改用户目录下的 `.condarc` 文件来使用 TUNA 镜像源。Windows 用户无法直接创建名为 `.condarc` 的文件，可先执行 `conda config --set show_channel_urls yes` 生成该文件之后再修改。

注：由于更新过快难以同步，我们不同步 `pytorch-nightly`，`pytorch-nightly-cpu`，`ignite-nightly` 这三个包。

创建激活新的环境

```
(base) xb23@bbt:~$ conda create -n pro
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

```
==> WARNING: A newer version of conda exists. <==
current version: 4.8.2
latest version: 24.3.0
```

Please update conda by running

```
$ conda update -n base -c defaults conda
```

Package Plan

environment location: /rd1/home/LEB2024/xb23/miniconda3/envs/pro

Proceed ([y]/n)? y

```
(base) snq23@bbt:~$ conda activate pro
(pro) snq23@bbt:~$ conda --help
```

conda env remove 删除名为 pro 的环境

安装 hmmer 工具包

[:: Anaconda.org](https://anaconda.org) 可以搜索需要安装的包是否存在

```
(pro) xb23@bbt:~$ conda install hmmer -c bioconda
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

The following NEW packages will be INSTALLED:

hmmer bioconda/linux-64::hmmer-3.1b2-3

Proceed ([y]/n)?

The screenshot shows the Anaconda.org search results for the package 'hmmer'. The search bar at the top contains 'hmmer'. Below the search bar, there are filters for 'Type: All', 'Access: All', and 'Platform: All'. The results table lists several packages, including 'bioconda / hmmer 3.4', 'bioconda / pyhmmer 0.10.11', 'bioconda / perl-bio-searchio-hmmer 1.7.3', and 'bioconda / hmmer2 2.3.2'. The 'hmmer 3.4' package is highlighted.

⬆ Favorites	⬆ Downloads	⬆ Artifact (owner / artifact)	Platforms
6	989251	bioconda / hmmer 3.4 Biosequence analysis using profile hidden Markov models	linux-64 linux-aarch64 osx-64
1	76849	bioconda / pyhmmer 0.10.11 Cython bindings and Python interface to HMMER3.	linux-64 linux-aarch64 osx-64
0	51166	bioconda / perl-bio-searchio-hmmer 1.7.3 A parser for HMMER2 and HMMER3 output (hmmscan, hmmsearch, hmmpfam)	noarch
0	20412	bioconda / hmmer2 2.3.2	linux-64 osx-64

conda remove 命令用来删除已经安装的工具

mamba

[Welcome to Mamba's documentation! — documentation](#)

mamba 是一个 C++ 语言实现的 conda，具有完全一样的功能且速度是 conda 的几十倍！
使用方式与 conda 完全一样，只需要把命令中的 conda 改为 micromamba
建议直接一开始直接使用 mamba