Exercise 1: Install and run Emu

(Note that you can view an outline of this document by clicking the icon that looks like a bullet point list, to the far right just above the header above)

Dependencies

This guide assumes you have access to a Linux environment, as highlighted in the preparations information.

To run Emu, you also need to set up a Conda or Pixi environment.

In this guide we will go with Pixi, as it is in our experience faster and more robust than Conda.

Setting up Emu from Bioconda with Pixi

Below is a guide on how to set up Emu through the bioconda repository, but using Pixi instead of Conda, for greater speed and flexibility.

1. Install Pixi

- The easiest way is to run the official install script from the webpage:

```
curl -fsSL https://pixi.sh/install.sh | bash
```

But there are alternative installation methods too.

2. Reload your shell, for example by executing bash again:

```
bash
```

3. Create a folder for your exercise with Emu:

```
mkdir -p ex1
cd ex1
```

4. Create a new pixi environment in this folder:

```
pixi init
```

This will create a file called `pixi.toml` inside the current folder.

5. Open the file in an editor and change the line:

   ```
   channels = ["conda-forge"]
   ```

   … to:

   ```
   channels = ["conda-forge", "bioconda"]
   ```

6. After saving the above change in `pixi.toml`, you can now run `pixi shell`, so that all installed software are available in your `$PATH` variable:

   ```
   pixi shell
   ```

   You can see that you have the pixi environment loaded by the `(ex1)` part added to the left in your terminal.

7. Now you can install Emu via Pixi:

   ```
   pixi add emu
   ```

8. Done! Now you should be able to check that Emu is installed, with:

   ```
   emu -h
   ```

   … and, for the central command:

   ```
   emu abundance -h
   ```

9. Before running on real data we also need to install the OSF Client in order to download the pre-built EMU database:

   ```
   pixi add osfclient
   ```

10. Then we can use the instructions in the Emu readme for downloading the default database via OSF. We need to create directory for it first though, and download it there, as the tar doesn't include one.

    ```
    mkdir emudb
    cd emudb
    osf -p 56uf7 fetch osfstorage/emu-prebuilt/emu.tar
    ```

    (The `-p` flag is for the project ID)

11. We unpack the tar, and move out of the `emudb` folder again:

```
tar -xvf emu.tar
cd ..
```

12. Done! Now we can run `emu abundance`, poinding to this `emudb` folder as the database, in the next step.

Run Emu on some test data

1. Download the test data tarball: Download the tar ball from this link and save it in the folder where you created the pixi environment above.

```
wget https://github.com/kclinmicro/16s-ont-
        course/releases/download/v0.0.1/16s-course-data.tar.gz
```

2. Unpack the tarball:

```
tar -zxvf 16s-course-data.tar.gz
```

3. You can look at how the file structure looks:

```
$ tree data
data
├── 16s_ont_example
│   ├── 16s_ont_example
│   │   └── 20260101_1200_X1_FBB32095_15aa3986
│   │       ├── fastq_pass
│   │       │   ├── barcode01
│   │       │   │   └──
        FBB32095_pass_barcode01_15aa3986_bbffde72_0.fastq.gz
│   │       │   ├── barcode02
│   │       │   │   └──
        FBB32095_pass_barcode02_15aa3986_bbffde72_0.fastq.gz
│   │       │   └── barcode03
│   │       │       └──
        FBB32095_pass_barcode03_15aa3986_bbffde72_0.fastq.gz
│   │       └── final_summary_FBB32095_15aa3986_bbffde72.txt
│   └── 16s_ont_example.csv
├── emu_data
│   ├── 16sneg.fq.gz
│   ├── 16spos.fq.gz
│   └── sample01.fq.gz
└── LICENSE.txt

8 directories, 9 files
```

We will use the three files in `data/emu_data` in this exercise.

4. Running Emu

   First check the options available to Emu's main command, `abundance`:

   ```
   emu abundance -h
   ```

   Then try to run Emu on the sample data with this command (make sure you understand what each flag does!):

   ```
   emu abundance \
     --db emudb \
     --type map-ont \
     --output-dir emuout \
     --threads 2 \
     --keep-files \
     --keep-counts \
     --keep-read-assignments \
     data/emu_data/sample01.fq.gz
   ```

5. Done! Now we can look at the abundance table in the next section.

Looking at Emu output

1. Explore the output files Now you should be able to look at the files generated:

   ```
   tree emuout
   ```

2. Look at the main output file.

   We start with changing directory to the `emuout` folder:

   ```
   cd emuout
   ```

   To view the main output, with the relative abundances, you can use this command:

   ```
   cat sample01.fq_rel-abundance.tsv | column -t -s $'\t' | less -S
   ```

   To think about: Why do you think we use the `column -t` command, and why do you think we supply a `TAB` charater to the `-s` flag of this command?

   And, why do we set the `-S` flag to `less`? How can you check this?

3. As you can see above, the table is not sorted by abundance. To create a sorted file we can use the GNU `sort` command:

```
cat sample01.fq_rel-abundance.tsv | sort -r -k 2,2 >
        sample01.abundance.tsv
```

And then we can look at this file with the command we used above:

```
cat sample01.abundance.tsv | column -t -s $'\t' | less -S
```

(Note: We don't strictly need to run `cat` at first, but could instead use `sort` or `column` directly, but starting with a simple `cat` makes it easier to re-use the commands from history (with the up-arrow on the termina) and modify it, based on what you want to see).

Summary

That's it for now. Now you know how to install and run Emu, and look at it's main output file.