Examples

Here we will create two conda packages, `czlab_perl_lib` and `mcross`, where `mcross` depends on `czlab_perl_lib`.

Create a conda packge for `czlab_perl_lib`

1. Create a folder named `czlab_perl_lib`

```
mkdir -p czlab_perl_lib
cd czlab_perl_lib
```

2. Create a file named `meta.yaml` under `czlab_perl_lib` with the following content

```
{% set version = "1.0.1" %}

package:
    name: czlab_perl_lib
    version: {{ version }}

source:
    url: https://github.com/zhixingfeng/czlab_perl_lib/archive/refs/tags/{{ version }}.tar.g
    md5: acc84ef3a33505066d36c12d5545968a

build:
    noarch: generic
    number: '0'
    string: czlab2021

requirements:
    run:
       - perl
       - perl-math-cdf
       - perl-bioperl
test:
    commands:
        - ls
about:
    home: https://github.com/zhixingfeng/mCross
    license: MIT
    summary: mCross Perl script
```

The most fields are self explained. Notably, `url` is the link to source code, typically a release of a GitHub repo. `noarch: generic` means platform independent since the source code is just perl script. `run:` under `requirements:` is the runtime dependences of the package: Perl and two Perl packages.

3. Create a file named `build.sh` under `czlab_perl_lib` with the following content

```
#!/bin/bash
mkdir -p $PREFIX/lib/czlab_perl_lib
cp -r ./* $PREFIX/lib/czlab_perl_lib

mkdir -p "${PREFIX}/etc/conda/activate.d"
mkdir -p "${PREFIX}/etc/conda/deactivate.d"

echo "P_BIOPERL=\$(dirname \$(dirname \$(find $PREFIX/lib -name SeqIO.pm)))" > "${PREFIX}/et
echo "export PERL5LIB=$PREFIX/lib/czlab_perl_lib:\$P_BIOPERL" >> "${PREFIX}/etc/conda/activa

echo "unset PERL5LIB" > "${PREFIX}/etc/conda/deactivate.d/env_vars_czlab_perl_lib.sh"


chmod u+x "${PREFIX}/etc/conda/activate.d/env_vars_czlab_perl_lib.sh"
chmod u+x "${PREFIX}/etc/conda/deactivate.d/env_vars_czlab_perl_lib.sh"
```

`PREFIX` is the conda build-in environment variable describing the directory of the conda environment, where the package is installed. The current folder, `./`, is the unzipped source code folder, i.e., the unzipped folder of the tar ball in `url`. `activate.d` and 'deactivate.d' are two specially folders. All `.sh` files in `activate.d` will be executed when the environment is activated (`conda activate <environment name>`), and all `.sh` files in `deactivate.d` will be executed when the environment is deactivated (`conda deactivate`). we set `PERL5LIB` in `env_vars_czlab_perl_lib.sh` to add additional library searching path. **It is important to note that** `PREFIX` is only viable when `build.sh` is executed during package building phase and will be unset afterward, so it is can NOT be used to set environment variable. You should ignore the `activate.d` and `deactivate.d` if you have no environ

4. Before building the package, you have to add additional conda channels, `conda-forge` and `bioconda` if you haven't done so before, because Perl and Perl packages are not available in default channel. **Make sure that** `conda-forge` has higher channel priority than `bioconda`, otherwise it might cause package conflict errors.

```
conda config --append channels conda-forge
conda config --append channels bioconda
```

5. Set conda channels priority to `strict` to increase build speed and reduce the risk of package conflict. See more details about conda channel priority.

```
conda config --set channel_priority strict
```

6. Install `conda-build` if you haven't done this before.

```
conda install conda-build
```

7. Login your anaconda account if you haven't done this before.

```
anaconda login
```

8. Build and upload your package to https://anaconda.org/.

```
cd ..
conda build czlab_perl_lib
```

If the package is not uploaded to https://anaconda.org/ automatically, follow the instruction in the output of `conda build`.

9. Search `czlab_perl_lib` in https://anaconda.org/, you should find the package if everything works. Click the package and you will find the installation instruction.

Create a conda packge for `mcross`, which depends on `czlab_perl_lib`

1. Create a folder named `mcross`

```
mkdir -p mcross
cd mcross
```

2. Create a file named `meta.yaml` under `mcross` with the following content

```
{% set version = "0.9.5" %}

package:
    name: mcross
    version: {{ version }}

source:
    url: https://github.com/zhixingfeng/mCross/archive/refs/tags/{{ version }}.tar.gz

build:
    noarch: generic
    number: '0'
    string: mc2019

requirements:
    run:
       - czlab_perl_lib
       - bioconductor-motifstack
       - bioconductor-limma
       - r-base
       - r-getopt
       - r-ggplot2
       - r-gridextra
       - r-cowplot
test:
```

```
    commands:
       - ls

about:
    home: https://github.com/zhixingfeng/mCross
    license: MIT
    summary: The script to detect cross linking sites
```

3. Create a file named `build.sh` under `mcross` with the following content

```
#!/bin/bash
mkdir -p $PREFIX/bin
cp -r ./* $PREFIX/bin
```

4. Build and upload your package to https://anaconda.org/.

```
cd ..
conda build mcross
```

Install mcross

```
conda create -n mcross
conda activate mcross
conda install -c <your channels name> mcross
mCross.pl
```

You will get the following message if everything works

```
Motif discovery anchored by crosslink sites ...
Usage: mCross.pl [options] <seq_file> <out_file or out_file_stem>
 -l       [int]   : sequence extension around crosslink site (10)
 --seed   [file]  : top_nmer_file
 --bg     [file]  : if top_nmer not provided, fg and bg file are used to get the list
 -p       [int]   : pad the seed motif on both sides (0)
 -m       [int]   : number of mismatches allowed in the core motif (1)
 -N       [int]   : max number of seed words to search (20)
 --cluster-seeds  : cluster seed word
 --xl-model     [int]   : crosslink model (1=simple(default), 2=nucleotide-specific)
 --score-method [string]: [log]|sqrt
 --prefix       [string]: prefix of the motif name (RBP)
 --single-output-file   : write all motifs to a single file
 -c             [string]: cache dir (./mCross.pl_1638772969_0.880516937309043)
 -v                     : verbose
```

Important notes for creating conda package written in script language

1. Make sure you add a generic Shebang for each script file. for example, you should use

```
#!/usr/bin/env perl
```

do NOT use

```
#!/usr/bin/perl
```

2. Make sure you are using the correct interpreter, which should located in the conda environment folder. For example

```
which perl
```

the output should be something like

```
~/work/tools/anaconda3/envs/mcross/bin/perl
```

NOT

```
/usr/bin/perl
```