p_batch tutorial

Run multiple proteins concurrently with MCCE

Multiple PDB MCCE Runs (pro_batch)

p_batch is a program included in MCCE, under the folder MCCE_bin. It accepts a directory containing ".pdb" files, and runs MCCE with identical settings on each protein file. Usage: p_batch protein_dir [custom_script.sh] By default, default_script.sh is created by p_batch at run time, if it does not exist. The default scripts calls for steps 1-4 of MCCE to be run at level 1, assumes a dielectric constant of 8 with NGPB as the Poisson Bolztmann solver, and sets entropy control on. Another tool is available called "pro_batch". This behaves very similarly to p_batch, though the file pro_batch creates by default has more features for heavy users.

Custom MCCE Runs and submit_shell

Default MCCE runs (run_mcce4)

By default, each step of MCCE has certain parameters set to it. For example, step 3 defaults to the Gunner Lab's in-house Poisson-Boltzmann solver, NGPB, though Delphi and ZAP are available to those with the respective OpenEye licenses. But what if the user wants non-default settings?

By default, run_mcce4 uses the following settings:

```
step1.py {input_pdb} -d 4 --dry
step2.py -l 1 -d 4
step3.py -d 4
step4.py --xts -i 7 -n 1
```

Each of these lines is a shell command. If you were to copy and paste these lines (after changing {input_pdb} to a local PDB file), MCCE would run just as though "run_mcce4" had been the command. To see more of our options, let's look at the output of "step1.py -h":

```
step1.py -h
usage: step1.py [-h] [--norun] [--noter] [-e /path/to/mcce] [-u Key=Value] [-d epsilon] [-
load_runprm prm_file] [--dry] pdb

Run mcce step 1, premcce to format PDB file to MCCE PDB format.

positional arguments:
  pdb

options:
  -h, --help            show this help message and exit
  --norun               Create run.prm but do not run step 1; default: False.
  --noter               Do not label terminal residues (for making ftpl); default: False.
  -e /path/to/mcce      mcce executable location; default: mcce.
  -u Key=Value          User customized variables; default: .
```

```
  -d epsilon            protein dielectric constant for PBE solvers; default: 4.0.
  -load_runprm prm_file
                        Load additional run.prm file, overwrite default values.
  --dry                 Delete all water molecules; default: False.
```

So, if we wanted to start step 1 with an alternate version of MCCE, a dielectric constant of 8, and keeping explicit water molecules, we could change the command to:

```
step1.py {input_file.pdb} -e /home/other_mcce -d 8
```

Following a similar process for each step, we can create a full custom run to our liking.

submit_mcce4.sh

However, it's tedious to submit these instructions over and over. In MCCE4-Alpha/schedulers, a file named "submit_mcce4.sh" can be found. We recommend copying "submit_mcce4.sh" to the local directory and customizing it to fit the desired job. Let's look at the script:

```
# Input and Output:
input_pdb="prot.pdb"    # (INPDB)

# Set MCCE4 Parameters
MCCE_HOME="/home/MCCE4-Alpha"
USER_PARAM="./user_param"
EXTRA="./user_param/extra.tpl"
TMP="/tmp"
CPUS=1

# Step control flags
step1="t"               # STEP1: pre-run, pdb-> mcce pdb  (DO_PREMCCE)
step2="t"               # STEP2: make rotamers            (DO_ROTAMERS)
step3="t"               # STEP3: Energy calculations      (DO_ENERGY)
step4="t"               # STEP4: Monte Carlo Sampling     (DO_MONTE)
step_clean="t"          # Clean PBE data                  (BACKUP CLEAN) : Set to f if step3
--debug option is used

# Optional step controls
stepM="f"               # Generate Partial Membranes                    : If true, user MUST
satisfy condidtions of stepM.sh, which can be be obtained on MCCE4/inhouse/stepM.sh
stepA="f"               # Run a custom script between step1 and step2   : If true, user MUST
```

```
satisfy condidtions of their custom script
stepB="f"               # Run a custom script between step2 and step3   : If true, user MUST
satisfy condidtions of their custom script
stepC="f"               # Run a custom script between step3 and step4   : If true, user MUST
satisfy condidtions of their custom script

# MCCE Simulation
STEP1="step1.py -d 4 --dry"
STEP2="step2.py -d 4 -l 1"
STEP3="step3.py -d 4 -s delphi -p $CPUS -t \$TMP"
STEP4="step4.py --xts -i 7 -n 1"

# Optional MCCE script locations
STEPM="/path/to/stepM.sh"          # Optional StepM: Bash script
STEPA="/path/to/stepA_script.py"   # Optional StepA: Python script to run between step1 and
step2.
STEPB="/path/to/stepB_script.py"   # Optional StepB: Python script to run between step2 and
step3.
STEPC="/path/to/stepC_script.py"   # Optional StepC: Python script to run between step3 and
step4.

# NOTE: User is responsible to precheck if custom scripts work properly and efficiently
```

- "Set MCCE4 Parameters" controls which version of MCCE is being used.
- "Step control flags" allow for steps to be turned on and off between runs.
- "Optional step controls" allow for custom scripts to be run between steps, and partial membrane generation
- "MCCE Simulation" is where each step is customized- use "step[1-4].py -h" to see options for each step
- Lastly, the paths to any custom scripts are referenced

Then, submit_mcce4.sh can be executed:

```
chmod +x submit_mcce4.sh # make this file an executable
./submit_mcce4.sh # execute the submit shell
```

We recommend using custom submit shells in conjunction with p_batch/pro_batch. These programs included with MCCE4-Alpha are able to accept a -custom flag and a local shell script, to execute it across all relevant jobs. Learn about this functionality here.

How do I run multiple proteins at once? p_batch and pro_batch

p_batch is a program included in MCCE, under the folder MCCE_bin. It accepts a directory containing ".pdb" files, and runs MCCE with identical settings on each protein file.

```
p_batch -h

usage: p_batch [-h] [-custom script_path] input_path

p_batch accepts a directory containing PDB files, and executes identical MCCE runs in
directories named after each respective PDB file.

positional arguments:
  input_path           Path to a .pdb file

options:
  -h, --help           show this help message and exit
  -custom script_path  Give a shell script with custom instructions. If not defined, a default
script will be created and used.
```

By default, default_script.sh is created by p_batch at run time, if it does not exist. The default scripts calls for steps 1-4 of MCCE to be run at level 1, assumes a dielectric constant of 8 with NGPB as the Poisson Bolztmann solver, and sets entropy control on.

Another tool called "pro_batch" is available. This behaves very similarly to p_batch, though the default instruction script pro_batch creates has more features for heavy users, including the ability to use a scheduler. Both p_batch and pro_batch can accept custom shell scripts with the "-custom" flag. Learn about creating and editing shell scripts here.

```
pro_batch -h
```

```
usage: pro_batch [-h] [-custom script_path] [--sbatch | --no-sbatch] input_path

pro_batch accepts a directory containing PDB files, and executes identical MCCE runs in
directories named after each respective PDB file. pro_batch creates
a high-level shell script. The user then edits the shell script to their liking, and executes
it with the -custom flag.

positional arguments:
  input_path            Path to a directory containing PDB files.

options:
  -h, --help            show this help message and exit
  -custom script_path   Give a shell script with custom instructions. If not defined, a
default script will be created and used.
  --sbatch, --no-sbatch
                        Turn on this flag to use a scheduler. (default: False)
```

Let's look at an example of how p_batch can be used. First, create a directory, and fill it with protein files. For this example, assume a directory called "source_files" containing a couple PDB files.

```
user@example:/pro_batch_testing$ ls source_files
1ans.pdb  4pti.pdb
```

Now, let's use p_batch.

```
p_batch source_files

New book.txt created.  You can remove protein files to be run by editing book.txt if desired,
and resume by running p_batch again.

These proteins will be run:

4pti
1ans
Pre-existing directories for these proteins will be emptied and replaced with information from
the new run.
Run MCCE with the current settings? (yes/y/no)
```

Typing "yes" or "y" will start the default MCCE process in each directory (unless the -custom flag is used to choose an alternate script).

```
Processing source_files/4pti.pdb...
Processing source_files/1ans.pdb...

Bash script is being executed in each directory. Double check processes are running with
command 'top', or 'mcce_stat'. mcce_stat also updates book.txt to reflect completed runs.
```

mcce_stat

Reviewing multiple protein runs can be cumbersome. To aid the user, "mcce_stat" is included. The directories created by p_batch or pro_batch will contain the output files of each MCCE run. mcce_stat checks of these directories for "signal" files to check how each run is progressing, as of mcce_stat's runtime.

```
mcce_stat

To see when PDB was run, reference mcce_timing.log in running directory. (pro_batch)

Completion: 7.50%
Directory step1         step2         step3     step4  Status
1ans      Exists        Exists
4pti      Exists
```

When all four steps are completed, the pKas of the selected protein will be available to see in "pK.out".

book.txt

Before beginning the MCCE run, the program creates a file named "book.txt". book.txt is a text file containing a list names of proteins that can be edited to "turn off" proteins. This is what our new book.txt file looks like:

```
4pti
1ans
```

If we wanted to turn off both of these, we could edit book.txt to:

```
4pti   x
1ans   c
```

Where "x" means exclusion, and "c" means completed. Attempting to run p_batch now results in the following message:

```
p_batch source_files/

book.txt found! Protein files identified in book.txt:

4pti   x
1ans   c

No runnable proteins found. Remove 'c' or 'x' from lines in book.txt to make them runnable,
add protein files to run, or delete book.txt to re-do all runs.

Aborting MCCE...
```

In this way, the user can choose any desired subset of proteins from the source directory's protein set.