GROMACS - Task #3357

Make sure "colvars" interoperates stably and smoothly with gromacs

01/28/2020 11:04 AM - Christian Blau

Status:	New	
Priority:	Normal	
Assignee:		
Category:		
Target version:		
Difficulty:	uncategorized	

Description

The Collective Variables Interface (Colvars) provides a huge number of algorithms that are based on collective variables.

To make sure colvars works smoothly with GROMACS we should work to establish a stable interface that allows to

```
- access system information over the course of the simulation efficiently
 - add to forces, virial, energy
 - hook up colvars to gromacs easily
```

A working version is stored here:

https://github.com/Colvars/colvars/pull/190

History

#1 - 02/05/2020 01:57 PM - Hubert Santuz

Hi,

I'm starting to re-write our interface Colvars-Gromacs to fit into the new MDModule machinery and I have a couple interrogations.
Colvars works by reading a separate configuration file, so 1) we don't need to modify the mdp file but 2)we need to add a new input file to mdrun.

About 1), I have implemented a class herited from IMDModule :

```
class Colvars final : public IMDModule
{
...
}
```

but from I understand, I still need to implement mdpOptionProvider() (due to the virtual declaration in IMDModule) even though I don't need to modify the mdp options, right?

About 2), currently, it's not possible to add a custom parameters to mdrun for a new file (for example gmx mdrun -colvars file.dat). We still have to modify legacymdrunoptions.h to hardcode the keyword?
Secondly, the only class we can use is IMDOutputProvider, there is no IMDInputProvider. Is this something you consider to add?

I'm not sure I'm posting to the right place so feel free to point me to a better resource if needed.

#2 - 02/05/2020 02:40 PM - Paul Bauer

Hello Hubert, thanks for the feedback.
I think you can have a look at the code Christian Blau wrote for the density fitting to see how to read in files during preprocessing.
We want to move away from people having to declare extra files at the invocation of mdrun (as it means that we can't have the whole simulation defined in the tpr, affecting portability to different systems).

This means that the colvars information should also be part of the TPR in the end.

#3 - 02/05/2020 05:29 PM - Giacomo Fiorin

Hello Paul, thanks for clarifying this important detail. This is a reasonable choice.

However, it would be very tricky to input all Colvars configuration parameters as individual MDP parameters for two reasons: (1) they follow a different format with its own parser, and (2) there are many Colvars keywords, not all of which are used very often and they would needlessly bloat the MDP doc page at http://manual.gromacs.org/current/user-guide/mdp-options.html.

Do you have an example that Hubert can use for embedding a text file into the TPR, consistent with the architecture of future GROMACS releases?
From a quick look, all modules that apply external forces seem to use MDP parameters, with the exceptions of:

- AWH (uses numbered awhinit?.xvg files as an alternative to the -awh command line option, which may be phased out at some point if I understand correctly).
- Density fitting, which uses MDP parameters alongside a filename for the density file itself, which is probably too large to be embedded in the TPR (please correct me if appropriate).

Would a MDP parameter along the lines of colvars-configuration-filename work?

#4 - 02/07/2020 01:04 PM - Paul Bauer

I added Christian Blau to the discussion, so I can clarify things about the modules.

#5 - 02/27/2020 10:53 PM - Erik Lindahl

We will gradually move towards hierarchical input files (and replacing mdp text files with YAML at some point).

However, user inputs should be parsed when preparing the input (i.e., when running grompp) - we don't want to introduce a second-stage text file parsing that only happens at runtime.

#6 - 03/04/2020 11:25 AM - Christian Blau

When the MdModules are called during grompp-time, they get a chance to do the file parsing when they receive the callback to store their internal parameters.

I suggest colvars subscribes to the preProcessingNotifications_ KeyValueTreeObjectBuilder that allows colvars to parse it's inputs and stores it as "internal" parameters in a key value tree that then may be read from the module
signing up to the simulationSetupNotifications_ KeyValueTreeObject.

The easiest way is just dumping the complete input file as a string in a single KVT tree entry; then the .tpr would work as a plain container that just stores the colvars input file.