KEPLER AMBER GPU MOLECULAR DYNAMICS WORKFLOW

USER MANUAL

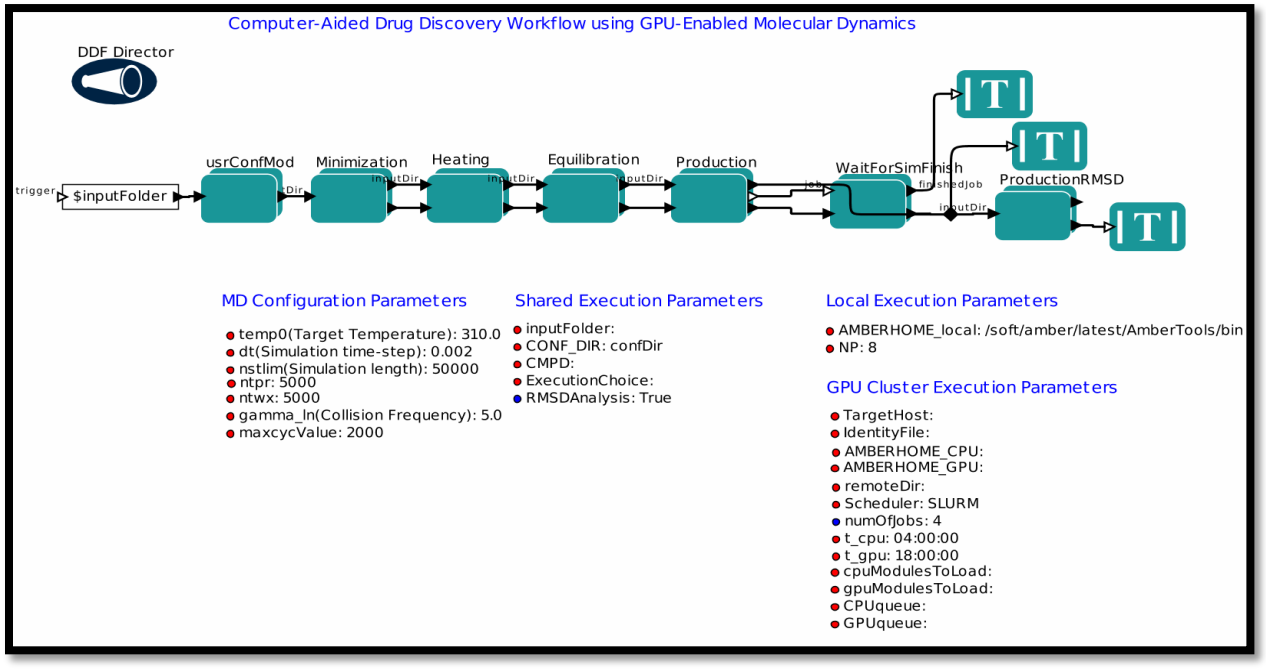


TABLE OF CONTENTS

Chapter 1: Introduction and Tool Description

Molecular dynamics (MD) simulations are electronic microscopes that allow researchers to study atomic details of biological systems (1). Applying Newton's laws of motion, MD simulations can predict atom movements following a time series. MD simulations usually involve four steps: minimization, heating, equilibration, and production, and the purpose of this Amber GPU MD simulation workflow is to automate these processes using graphics processing units (GPUs). This workflow can be used for all-atom molecular dynamic simulations, which involve five steps of minimization, one step of heating, three steps of equilibration, and one or more instances of production. MD simulations simulate atomic movements to help scientists predict reaction mechanisms, identify druggable pockets, and understand allosteric regulation (1). Moreover, due to the simple nature of this workflow, the workflow can be used as a starting point for learning how to run MD simulations. It can also be used a starting point developing workflows that involve more complicated aspects MD. It is our hope that this workflow will enable improved dissemination and reproducibility in the field of molecular dynamics.

Chapter 2: Installation and Program Requirement

The MD workflow requires users to have Kepler, bioKepler 1.2, R, and Amber installed on their local machines (2-4). Users can find more information about Kepler and bioKepler 1.2 at [http://www.biokepler.org/install-biokepler-1.2], R at [https://www.r-project.org/] and Amber at [http://ambermd.org/].

1. *Installing Kepler:*
   Users can download the latest bioKepler software from the following link:
   https://kepler-project.org/users/downloads

   To install on **Windows**, double click on the executable Kepler-2.5-win.exe.
   To install on **Mac OS X**, mount the Kepler-2.5.dmg by double-clicking on it and dragging-and-dropping the Kepler-2.5 folder into the /Applications folder.
   To install on **Linux**, simply untar Kepler-2.5-linux.tar.gz at a location of your choice, which will generate a folder named Kepler-2.5. This is the command line command to untar a tar.gz file:

   ```
   tar -C /[myfolder] -zxvf Kepler-2.5-linux.tar.gz
   ```

   For the following tutorial, we will work on a Linux platform as an example. Windows and Mac OS X installations follow similar steps.

2. *Starting Kepler:*
   Move into the Kepler installation directory.

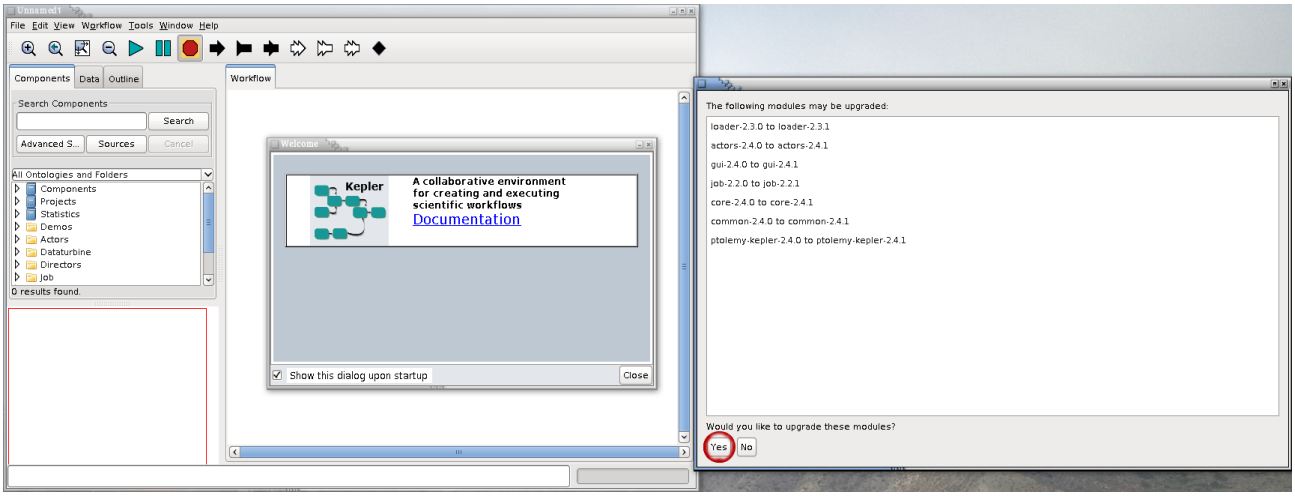
   ```
   cd /[myfolder]/Kepler-2.5
   ```

   To start the Kepler GUI, execute the kepler.sh script.

   ```
   kepler.sh
   ```

A pop up window may prompt for first time users to upgrade several modules, click Yes (red circle in Fig. 1), and Kepler will automatically upgrade and restart.

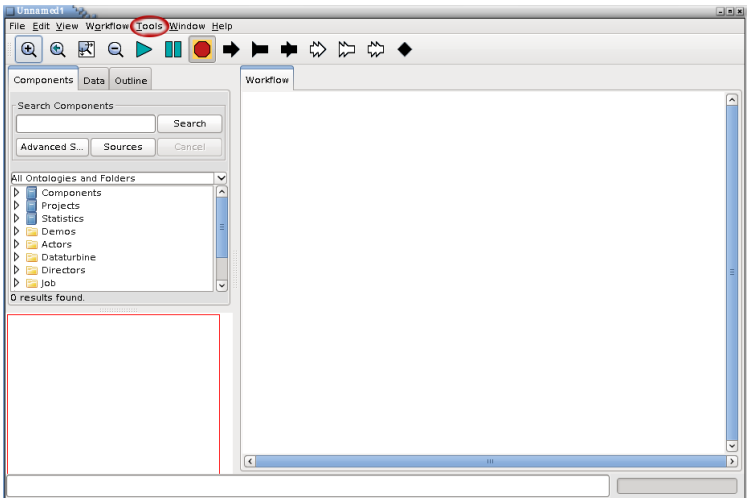
Fig. 1



3. *Installing bioKepler:*
   Our workflow allows users to choose between running the MD simulations with either a GPU cluster or a local GPU card. To allow this flexibility in a single workflow, we utilize a module from bioKepler. Hence, besides installing the Kepler software, we need to also install the bioKepler module.

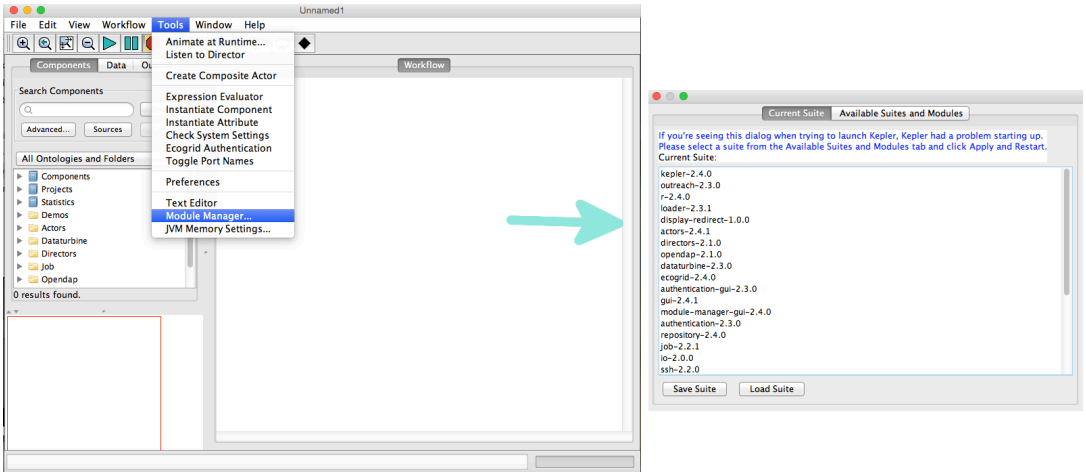
   The Kepler GUI will reopen after the modules are successfully updated (Fig. 2). Go to the menu bar on the top left of the screen and select Tools (Fig. 2, red circle).

Fig. 2



From the drop down menu under Tools, select Module Manager and a new window will pop-up (Fig. 3).

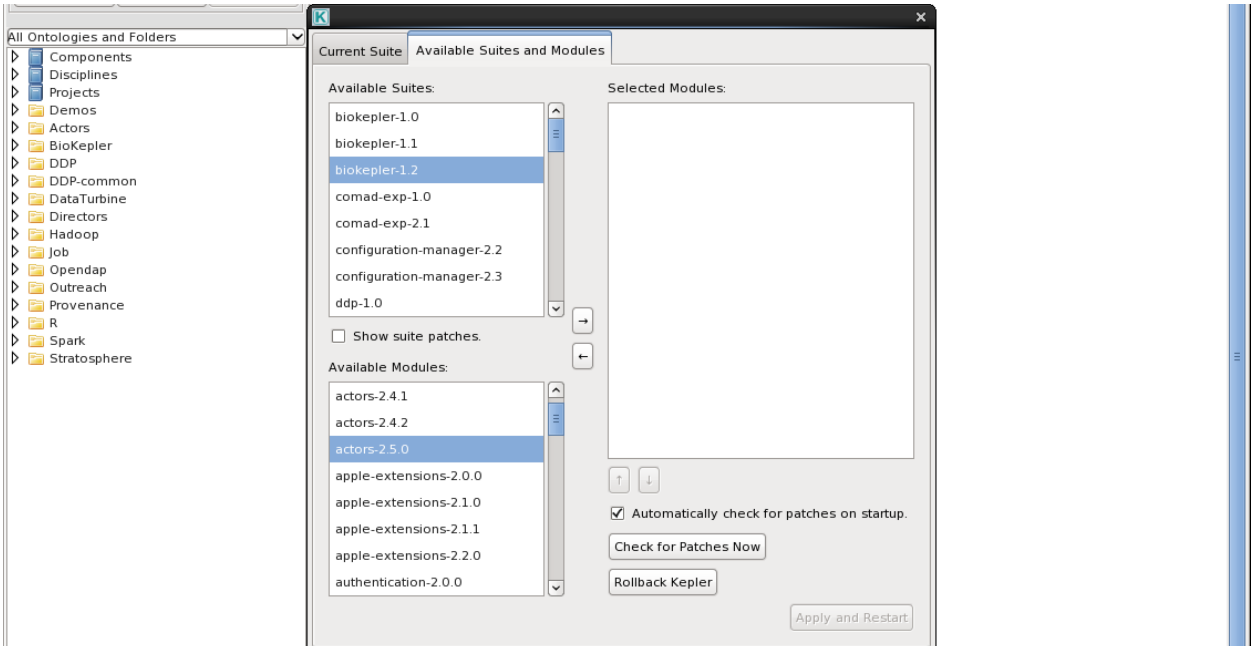
Fig. 3



There are two tabs at the top of the new window. Click on Available Suites and Modules, which will display a list of available suites and modules. Among the Available Suites, select bioKepler 1.2, and then click on the right arrow to transfer bioKepler 1.2 to the Selected Modules window (Fig. 4).

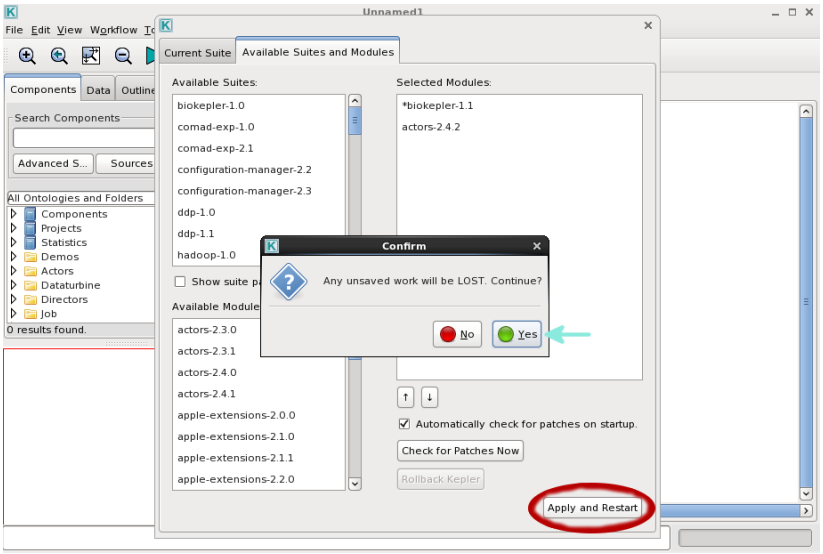
Do the same thing for the actors-2.4.2 modules. Among the Available Modules, select actors-2.5.0, and then click on the right arrow to transfer actors-2.5.0 to the Selected Modules window (Fig. 4).

Fig. 4



Finally, apply the selected modules, and restart Kepler by clicking on the Apply and Restart tab at the bottom right corner of the window (Fig. 5). Kepler will warn users that any unsaved work will be disregarded. Please save any unsaved work, and click Yes (Fig. 5).

Fig. 5



The Kepler GUI will reopen after successfully installing bioKepler. You can close it because we will run the MD workflow using command prompt.

4. *Downloading the MD workflow:*

The MD workflow is prepared to run on the Kepler platform. The workflow itself is a product developed by NBCR (http://nbcr.ucsd.edu/wordpress2/) and WorDS (http://words.sdsc.edu/).

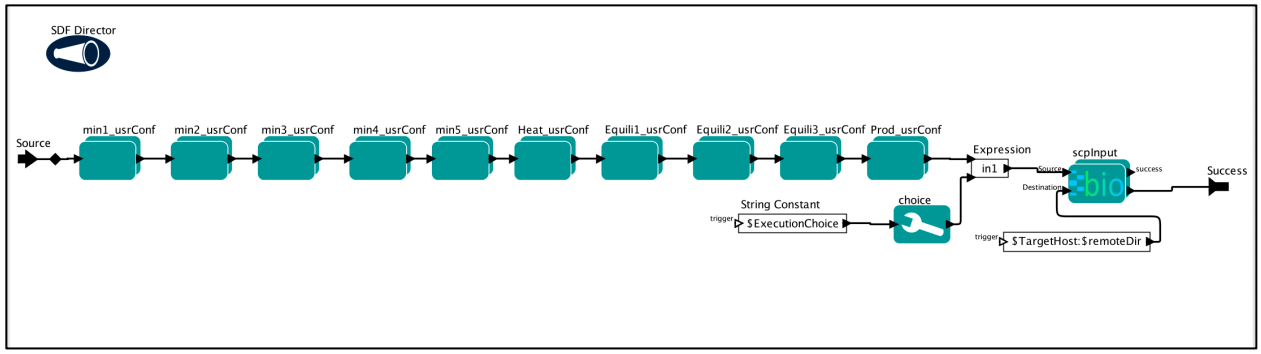
The MD workflow, this manual, and sample inputs can be downloaded from github or from the following link: (https://github.com/nbcrrolls/workflows/tree/master/Production/AmberMDSimulationGPU). If the workflow is not found in the above link, users can go directly to the NBCR website (http://nbcr.ucsd.edu/). Under the Download tab on the home page (http://nbcr.ucsd.edu/?page_id=3912), click on "Amber GPU MD" and it will take user directly to the github repository, where the workflow (AmberGPUMDSimulation) is located.

Chapter 3: Actor Descriptions

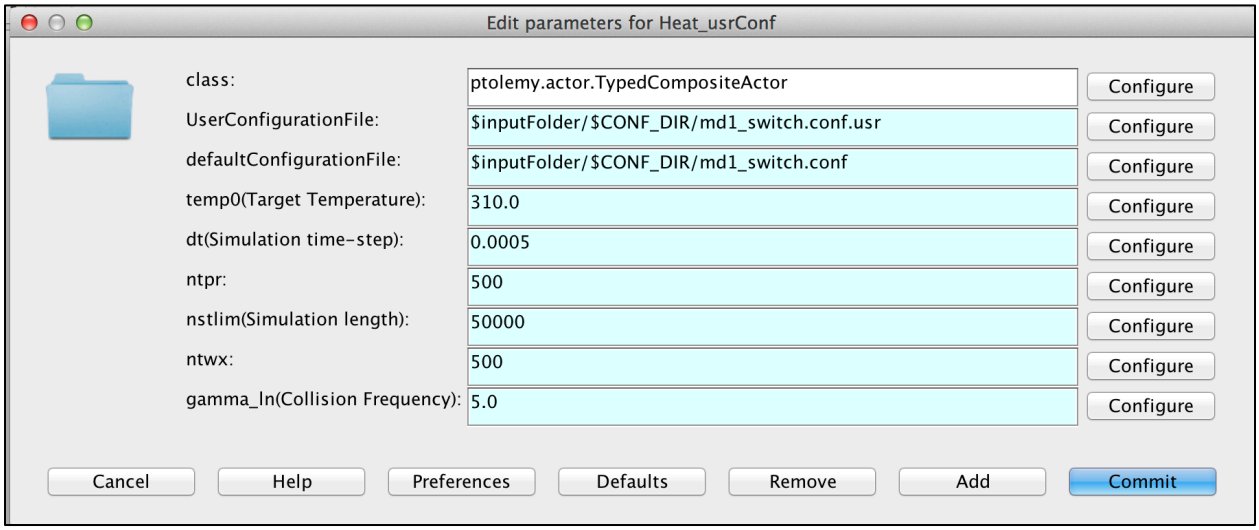
This chapter contains a description of each actor and its inputs, outputs, and runtime options. The description of each actor should also describe the component actors.

There are six main components to this workflow: usrConfMod, Minimization, Heating, Equilibration, Production, and WaitForSimFinish.

The **usrConfMod actor** module is a composite actor. It enables user to configure MD-parameters for minimization, heating, equilibration and production steps through command-line or GUI interface. Please see the source code for the template input files and check out what parameters users can configure through Kepler. If there are more parameters that users would like to change or users are interested in running nonconventional MD, it is advised that users put in their own script in the confDir file before starting the workflow. The module contains user configuration actors for four or five steps of minimization, one step of heating, three steps of equilibration and a step of production stage to individually control MD parameters for each step.



For heating, equilibration and production steps users can configure target temperature, simulation time-step, ntpr, simulation length and others. For minimization steps users can control maxcycValue through parameters.



The **Minimization actor** executes five minimization steps, bringing the system to an energetically relaxed state. Each minimization step has the option to either run locally on CPUs or run on the CPUs of a remote GPU cluster. Currently, the remote GPU cluster option is configured to submit jobs using the SLURM scheduler, but the user can also configure the workflow to fit their needs with the GenericGPUCluster option (see chapter 5). The Minimization actor also contains an analysis function that extracts the system energy data from the minimization output files and plots the energy against the number of time steps. Furthermore, the analysis function also creates an overall, serialized graph of the energy data from all five minimizations.

The **Heating actor** executes one heating step. The purpose of this actor is to heat the system from 0 K to the user's desired temperature. The Heating actor is configured to either run locally on a single GPU card or remotely on a remote GPU cluster. This actor also contains two analysis functions: an RMSD analysis and a combined energy, pressure, and temperature analysis. The workflow extract RMSD, energy, pressure, and temperature data. Energy, pressure, and temperature data are plotted over steps. The RMSD analysis is completed through cpptraj (2), a trajectory analysis program within Amber. The data are then sent to an R actor (5) and plotted against the number of time steps for RMSD and against the simulation time evolution for energy, pressure and temperature.

The **Equilibration actor** executes three equilibration steps that allow the system to reach an equilibrium state that is ready for MD simulation. Like the Heating actor, it also contains the RMSD, energy, pressure, and temperature analysis, which the workflow will extract simulation data and plot them on R. The Equilibration actor is configured to either run locally or remotely on a remote GPU cluster.

The **Production actor** runs MD simulations, producing time-series trajectories describing the motion of a system in a user-specified state, such as NPT or NVT. This actor will run a serial simulation job on a local GPU card or submit one or more jobs in parallel to a remote cluster.

After the Production actor submits the jobs, the **WaitForSimFinish** actor checks the status of job submissions and waits for the submitted jobs to finish. This actor also copies all the input and output files from remote cluster back to the local machine. Within the actor, an RMSD function performs RMSD calculations from the MD trajectories generated by the Production actor. As with the Equilibration actor, RMSD analysis is done with cpptraj (6). The RMSD data are then plotted in R (5) against the number of time steps.

Chapter 4: Step-by-Step Tutorial

This chapter provides a step-by-step tutorial of how to run the workflow and review its output.

1. *Setting up input folders:*
   Before running the MD workflow, you need to set up and organize input folders in a specific required format. To run the MD workflow, users will need topology and coordinate files for the system of interest as well as Amber input scripts for minimizations and molecular dynamics (MD) simulations. Along with the workflow, there is a sample input directory called "inputs". Users can find both the topology and coordinate files and all the sample MD input scripts inside (Table 1). Please see the source code for the template input files and check out what parameters users can configure through Kepler. If there are more parameters that users would like to change or users are interested in running nonconventional MD, it is advised that users put in their own script in the confDir file before starting the workflow. Please be aware that the workflow will only recognize min1_switch.conf, min2_switch.conf, min3_switch.conf, min4_switch.conf, min5_switch.conf, md1_switch.conf (for heating), md2_switch.conf (for equilibration 1), md3_switch.conf (for equilibration 2), md4_switch.conf (for equilibration 3), md5_switch.conf (for production) as input file names. If users would like to use their own input scripts to run MD, they need to change their input file name to the corresponding Kepler recognized file name.

Table 1. Inside inputs:

p53_wt	p53_wt.top, p53_wt.crd
confDir	min1_switch.conf, min2_switch.conf, min3_switch.conf, min4_switch.conf, min5_switch.conf, md1_switch.conf, md2_switch.conf, md3_switch.conf, md4_switch.conf, md5_switch.conf

Download the files and create a new directory.

```
mkdir MD_TEST
```

Then, move the zip file into the new directory.

```
mv ~/Downloads/Input_Files.zip MD_TEST
```

Change directory into the MD_TEST folder, unzip the Input_Files.zip file and then remove it from the directory.

```
cd MD_TEST
unzip Input_Files.zip
rm Input_Files.zip
```

Two folders named confDir and p53_wt will now appear in the empty directory. In the confDir folder, we have all the necessary Amber input files including five steps of minimization (min1 to min5), one step of heating (md1), three steps of equilibration (md2

to md4), one 1ns of productions (md5) using the MD workflow. In the other folder named p53_wt, we have the topology and coordinate files. The folder name needs to match the stem name of the topology and coordinate file. For example, if the two files are named apple_tree.top and apple_tree.crd, the folder name will need to be apple_tree.

NOTE: All Amber input file scripts must be in the confDir directory with the extension .conf.

Now, you are ready to run the MD workflow for a single system. There are three modes of execution and we will go over each of them in the next section: **local execution**, execution on the **XSEDE Comet GPU cluster,** and execution on a **private GPU cluster**.

2. *Running the workflow using the Local Execution mode:*

In your local computer, go to the directory where Kepler was installed and where kepler.sh is located.

```
cd /PATH/kepler-2.4
```

And type the following command line in that directory to start the MD workflow. Users will need to modify the command based on their needs. The definition for each parameter is listed below. An **ABSOLUTE** path is need when running Kepler. If an absolute path is not specifically typed out in command prompt, Kepler won't be able to find the files or folders of interest and it will throw an error message.

The general command to run Kepler through command prompt is

```
./kepler.sh –runwf -nogui -parameters values /PATH/workflow.xml
```

Here is an example of how Kepler can be run for our test case:

```
./kepler.sh -runwf -nogui -ExecutionChoice LocalExecution -NP 8 -
inputFolder /home/user/MD_TEST -CMPD p53_wt -numOfJobs 3 -
AMBERHOME_local  $AMBERHOME/bin /PATH/ AmberGPUMDSimulation_1.1.xml
```

For more information about each parameter, please check the parameters (see section below).

Parameters:

Please customize the parameters below for the execution of your workflow:

ExecutionChoice: This parameter gives the option to run same workflow either on a local machine or in a GPU cluster.

For local execution: LocalExecution
For GPU cluster execution:

To run on the XSEDE SDSC Comet cluster: CometGPUCluster
To run on a private cluster, see chapter 5

NP: The number of CPU processors for the minimization steps. It is the same number of processors for each of the step. Users can customize this number based on the number of processors in their machine. For example: 8 will call 8 CPU processors.
CMPD: The folder name containing the topology and coordinate files. The folder name needs to match the stem name of the topology and coordinate file. For example: p53_wt
AMBERHOME_local: $AMBERHOME/bin on your local machine. This should be the path that points to where pmemd.MPI is located.
inputFolder: The inputFolder path on your local machine. It is the directory where the confDir and CMPD folder is located. An absolute path is required.
numOfJobs: This parameter will allow "n" production jobs to be submitted in series for a single system. For example, 3 means to run 3 copies of the same production, one after another.

3. *Running the workflow using the GPU Cluster Submission mode:*

Step 1. Creating id_rsa.pub files:

(Skip to step 3 if the user has already configured the SSH key with either the Amaro Lab GPU cluster or the XSEDE SDSC Comet GPU cluster.)

Most remote clusters require login authentications and passwords. In order to connect to the remote cluster through the command-line interface without having to type in password for each submission step, the easiest way is to generate an SSH key. To run Kepler from the command prompt smoothly, we will create SSH keys for logging into a remote cluster without password.

First, create an .ssh directory in your home directory in your local computer. If users already have an .ssh directory in their home directory, they can skip this step.

```
mkdir ~/.ssh
```

Then, in the .ssh directory, generate the public/private RSA key pairs with the below command line:

```
cd ~/.ssh
ssh-keygen -t rsa
```

The system will ask you the following questions:

Generating public/private rsa key pair.
Enter file in which to save the key (/Users/*you*/.ssh/id_rsa): *[Press enter]*
Enter passphrase (empty for no passphrase): *[Press enter]*
Enter same passphrase again: *[Press enter]*
Your identification has been saved in /Users/*you*/.ssh/id_rsa.
Your public key has been saved in /Users/*you*/.ssh/id_rsa.pub.
The key fingerprint is: *# 01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db (random codes)*

Step 2. Preparing for job submission in a remote cluster:

Log in to the remote cluster of your choice, for example, the Amaro Lab cluster or the XSEDE SDSC Comet cluster.

```
ssh username@gpu.amaro.ucsd.edu
```

or,

```
ssh username@comet.sdsc.edu
```

Type in the password. Go to the home directory and create an .ssh folder in the remote cluster. (The .ssh directory may already be there, which is fine; ignore this step).

```
cd
mkdir -p ~/.ssh
logout
```

Then, in your local computer, append the public key to your remote cluster.

```
cat ~/.ssh/id_rsa.pub | ssh username@comet.sdsc.edu 'cat >> ~/.ssh/authorized_keys'
```

or

```
cat ~/.ssh/id_rsa.pub | ssh username@gpu.amaro.ucsd.edu 'cat >> ~/.ssh/authorized_keys'
```

Now you should be able to log into the remote cluster from your local computer without needing to type your password.

Log in to the remote cluster and create a folder. This folder is the working directory in the remote cluster where users will run their simulation jobs

```
ssh username@comet.sdsc.edu
```

or

```
ssh username@gpu.amaro.ucsd.edu

cd
mkdir MD_TEST
```

You are ready to run the MD workflow.

Step 3.

1. *Running the workflow using XSEDE SDSC Comet GPU Cluster:*

In your local computer, go to the directory where Kepler was installed and where kepler.sh is located.

```
cd /PATH/WHERE/KEPLER/IS/INSTALLED/kepler-2.4
```

And type the following command line in that directory to start the MD workflow. Users will need to modify the command based on their needs. The definition for each parameter is listed below. An **ABSOLUTE** path for each file is need when running Kepler. If an absolute path is not typed out, Kepler won't be able to find the files or folders of interest and throw an error message.

The general command to run Kepler through command prompt is

```
./kepler.sh -runwf -nogui -parameters values /PATH/workflow.
```

Here is an example of how Kepler can be run for our test case.

```
./kepler.sh -runwf -nogui -ExecutionChoice CometGPUCluster -TargetHost
username@comet.sdsc.edu -inputFolder /home/user/MD_TEST -IdentityFile
/home/user/.ssh/id_rsa -remoteDir /home/username/MD_TEST -CMPD p53_wt -
numOfJobs 3 -AMBERHOME_CPU /opt/amber/bin -AMBERHOME_GPU
/shared/apps/gpu/amber -cpuModulesToLoad "module load amber" -
gpuModulesToLoad "module load cuda/6.5" -t_cpu "04:00:00" -t_gpu
"18:00:00" -CPUqueue "compute" -GPUqueue "gpu"
/home/user/AmberGPUMDSimulation_1.1.xml
```

Parameters:

Please customize the parameters below for your execution:

ExecutionChoice: This parameter gives the option to run same workflow either on a local machine or in a GPU Cluster. For job submission to the XSEDE SDSC Comet cluster, use CometGPUCluster.
TargetHost: The remote cluster login. For example: username@comet.sdsc.edu
inputFolder: The inputFolder path on your local machine where the confDir and compound files are located. A complete and absolute path is required.
IdentityFile: The identity file (id_rsa) that allows remote access. For example: /PATH/.ssh/id_rsa. A complete and absolute path is required.
CMPD: The folder name containing the topology and coordinate files. The folder name needs to match the stem name of the topology and coordinate file. For example: p53_wt
remoteDir: Work Directory on remote GPU cluster that was created in Step 2. For example: /home/username/MD_TEST. A complete and absolute path is required.
numOfJobs: This parameter will allow "n" production jobs to be submitted in parallel for a single system. For example, 3 means to run 3 copies of the same production.

AMBERHOME_cpu: $AMBERHOME on remote cluster on CPU nodes. For example, on some clusters, Amber package location for CPU nodes and GPU nodes is different. On the SDSC Comet cluster Amber executable on CPU nodes is located at /opt/amber/bin.
AMBERHOME_gpu: $AMBERHOME on remote cluster on GPU nodes. For example, on the SDSC Comet cluster the Amber executable on GPU nodes is located at /share/apps/gpu/amber.
cpuModulesToLoad: Modules dynamically modifies a user's execution environment to configure the shell for an application. The parameter "cpuModulesToload" prepares execution environment on CPU nodes. For example, SDSC comet requires amber module to be loaded (module load amber) on a CPU node to access pmemd.MPI in the execution environment. Accepted value is **"module load amber"**
gpuModulesToLoad: The parameter "gpuModulesToload" prepares execution environment on GPU nodes on a cluster. Accepted value is **"module load cuda/6.5"**
t-cpu: Time requests for each of the minimization step in the remote cluster. It should be greater than the time needed to finish each individual step and less than the maximum wall time.
t-gpu: Time requests for the heating step, equilibration steps and production step in the remote cluster. It should be greater than the time need to finish each individual step and less than the maximum wall time.
CPUqueue: The type of Comet (supercomputer) queue users submits their jobs.
GPUqueue: The type of Comet (supercomputer) queue users submits their jobs.

2. *Running the workflow using Private Amaro Lab GPU Cluster:*

The private Amaro Lab GPU cluster run mode serves as an example of how users can run their MD jobs in a private cluster of their choice. For more information on configuring the workflow to a generic GPU cluster, please refer to chapter 5.

On your local computer, go to the directory where Kepler is installed and where kepler.sh is located.

```
cd /PATH/WHERE/KEPLER/IS/INSTALLED/kepler-2.4
```

Type the command below in that directory to start the MD workflow. Users will need to modify the command based on their needs. The definition for each parameter is listed below. An **ABSOLUTE** path for each file is need when running Kepler. If an absolute path is not typed out, Kepler won't be able to find the files or folders of interest and throw an error message.
The general command to run Kepler through command prompt is

```
./kepler.sh -runwf -nogui -parameters values /PATH/workflow.
```

Here is an example of how Kepler can be run for our test case:

```
./kepler.sh -runwf -nogui -ExecutionChoice PrivateGPUCluster -
TargetHost username@gpu.amaro.ucsd.edu -inputFolder /home/user/MD_TEST
```

```
-IdentityFile /home/user/.ssh/id_rsa -remoteDir /home/username/MD_TEST
-CMPD p53_wt -numOfJobs 3 -AMBERHOME_CPU /cm/shared/apps/amber14/bin -
AMBERHOME_GPU /cm/shared/apps/amber14/bin -cpuModulesToLoad "" -
gpuModulesToLoad "" -t_cpu "01:00:00" -t_gpu "12:00:00" -CPUqueue
"highq" -GPUqueue "highq" /home/user/AmberGPUMDSimulation_1.1.xml
```

Parameters:

Customize the parameters below when you execute the workflow:

ExecutionChoice: This parameter gives the option to run same workflow either on a local machine or in a GPU Cluster. For job submission on the private Amaro Lab cluster: PrivateGPUCluster.
TargetHost: The remote cluster login. For example: username@gpu.amaro.ucsd.edu
inputFolder: The inputFolder path on your local machine where the confDir and compound files are located. A complete and absolute path is required.
IdentityFile: The identity file (id_rsa) that allows remote access. For example: /PATH/.ssh/id_rsa. A complete and absolute path is required.
CMPD: The folder name containing the topology and coordinate files. The folder name needs to match the stem name of the topology and coordinate file. For example: p53_wt
remoteDir: Work Directory on remote GPU cluster that was created in Step 2. For example: /home/username/MD_TEST. A complete and absolute path is required.
numOfJobs: This parameter will allow a user-specified number of production jobs to be submitted in parallel for a single system. For example, setting numOfJobs to 3 means to run 3 copies of the same production.
AMBERHOME_cpu: $AMBERHOME is path to amber14 package on a compute resource. Amber package locations on some clusters are different for CPU nodes and GPU nodes. $AMBERHOME_cpu is amber package location on CPU nodes for a cluster. For example, on the Amaro Lab cluster Amber executable on CPU nodes is located at /cm/shared/apps/amber14/bin
AMBERHOME_gpu: $AMBERHOME_gpu is amber package location on GPU nodes for a cluster. For example, on the Amaro Lab cluster Amber executable on GPU nodes is located at /cm/shared/apps/amber14/bin
cpuModulesToLoad: Modules dynamically modifies a user's execution environment to configure the shell for an application. The parameter "cpuModulesToload" prepares execution environment on CPU nodes. For the Amaro Lab cluster, the required modules are already loaded in environment by default hence no assignment is done for this parameter for Amaro Lab execution choice.
gpuModulesToLoad: The parameter "gpuModulesToload" prepares execution environment on GPU nodes on a cluster. For the Amaro Lab cluster, the required modules are already loaded in environment by default hence no assignment is done for this parameter for Amaro Lab execution choice.
t-cpu: Time requests for each of the minimization step in the remote cluster. It should be greater than the time need to finish the individual step and less than the maximum wall time.
t-gpu: Time requests for the heating step, equilibration steps and production step in the remote cluster. It should be greater than the time need to finish each individual step and less than the maximum wall time.

CPUqueue: The type of queue users submits their jobs.
GPUqueue: The type of queue users submits their jobs.

3. *Checking the MD simulation output data:*

Initial check

After about four hours, the simulations will be finished. Kepler will copy all the files back to the user's local machine. The files will be located in the inputFolder, inside the compound folder. Users can access them by changing the directory to the compound folder and listing all the files available.

```
cd /home/user/MD_TEST/p53_wt
ls
```

During the final production step, only one GPU can be used to run one production in the local machine. However, if the user runs the simulations using clusters, they are given the option to run multiple productions simultaneously. The production (md5) outputs will then be stored in separate folders under the compound folder, e.g., md5_Output1, md5_Output2, md5_Output3, etc.

Log files

When running jobs in remote clusters or supercomputers, the scheduler sometimes will generate some intermediate files or error messages. These files are saved in the _log directories for each simulation step: minimization (min1_log to min5_log), heating (md1_log), equilibration (md2_log to md4_log) and production (md5Log1, md5Log2, md5Log3, etc.)

Plots

The MD workflow plots energy against time step for the minimization steps, energy, pressure, temperature and root-mean-square-deviation (RMSD) for the heating, equilibration, and production steps. All the plots are saved in .png file formats and label as Source_and_type_of_plot.png. For example, the temperature plot from equilibration2 is labeled equil2_temp.png.

The minimization steps progressively minimize energy and release constraints, so the energy plots are good indicators whether a protein is actually minimized. If it is, we will see a gradual decrease of energy. Even though pressure and temperature fluctuate largely in the heating (md1) equilibration (md2 to 4), and production (md5) steps, the average is set to the user-defined pressure or temperature.

.rst files and .out files

One of the output files is the restart file. It is created if users need to restart the simulation from the save point. Every simulation step in the workflow generates a

restart file that is labeled *step*.rst, i.e., the minimization1 (min1) restart file is called min1.rst and the heating (md1) restart file is called md1.rst. Another type of output file is the .out file. This file outputs numerical data generated at each simulation step. It also prints out the initial inputs used to calculate the numerical outputs. Each simulation step generates an output file and is labeled with an .out extension. Users can read them with a text editor.

Trajectory files

Only the heating (md1), equilibration (md2 to 4), and production (md5) steps output trajectory files. No trajectories are produced from minimization. They are labeled with an .nc extension. The trajectory files depict dynamic protein motions with respect to time. They can be read using trajectory viewers such as VMD and Chimera (7, 8). Below, we present steps to read the files in VMD.

Reading files in VMD

Step 1.

Download VMD here

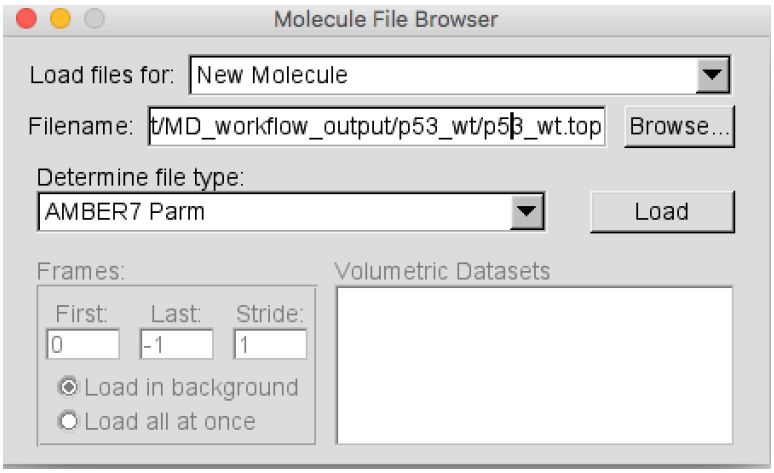
After installing VMD on the corresponding platform, click to open it. It's possible you will already have VMD installed. If so, please consult your IT administrator.

Step 2.

Load in topology file

After VMD is opened, users should see a display window and a main control panel. In the main control panel, click File -> New Molecule. Another screen will pop up. Click browse to choose file name. Go into the compound name folder and click the topology file (.top). Then choose AMBER7 Parm as from the Determine file type dropdown menu (Fig. 6). Click Load.

Fig. 6



Step 3.

Load in trajectory file

Then click browse again to choose the trajectory file. Do NOT choose Load files for to New Molecule. The file type should be automatically determined. Users do not need to change anything there. Then, click Load. A protein in water box should appear in the Display screen. For more information on how to use VMD, please check the VMD tutorial here.

Chapter 5: For Advanced Users

This chapter is for users who have finished the basic tutorial or are familiar with Kepler. The first section lay out the steps for configuring the workflow to GPU clusters other than the Comet cluster or the Amaro Lab cluster. The second section shows the steps for configuring the workflow to run more or fewer simulation steps. The third section shows how users can check the for workflow reproducibility in their results.

1. *Configuring the workflow to a generic GPU cluster*

Step 1.execution choice

Open the Kepler graphical user interface (GUI)

To configure the workflow so it will connect, submit, and monitor MD jobs on another GPU cluster, we will need to open Kepler GUI.

In your local computer, go to the directory where Kepler was installed and where kepler.sh is located. Then, start Kepler by typing kepler.sh

```
cd /PATH/WHERE/KEPLER/IS/INSTALLED/kepler-2.4
kepler.sh
```

Step 2.

Load the workflow into the GUI

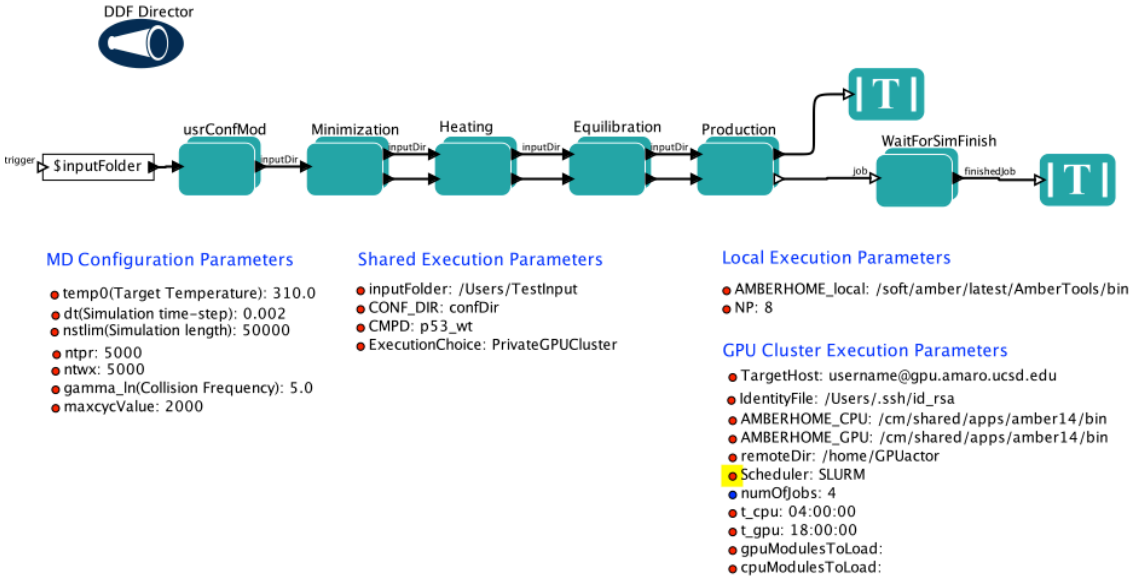
Go to File > Open, find the directory the workflow is in, and then click to load the MD workflow into the GUI.

Step 3.

Configure scheduler

After the workflow shows up in the GUI interface, double click on the scheduler parameter in the lower right corner of the screen underneath GPU Cluster Execution Parameters (Fig. 7) and change it to match the corresponding scheduler of the user's private cluster. For example, if the scheduler is PBS or SGE, change SLURM to PBS or SGE. Then, left click on the Minimization actor, then right click drop down a manual and left click on "open the actor".

Fig. 7



Step 4.

Configure individual actors

Double click on the minimization actor (Fig. 8) and a screen will pop up (Fig. 9).

Fig. 8

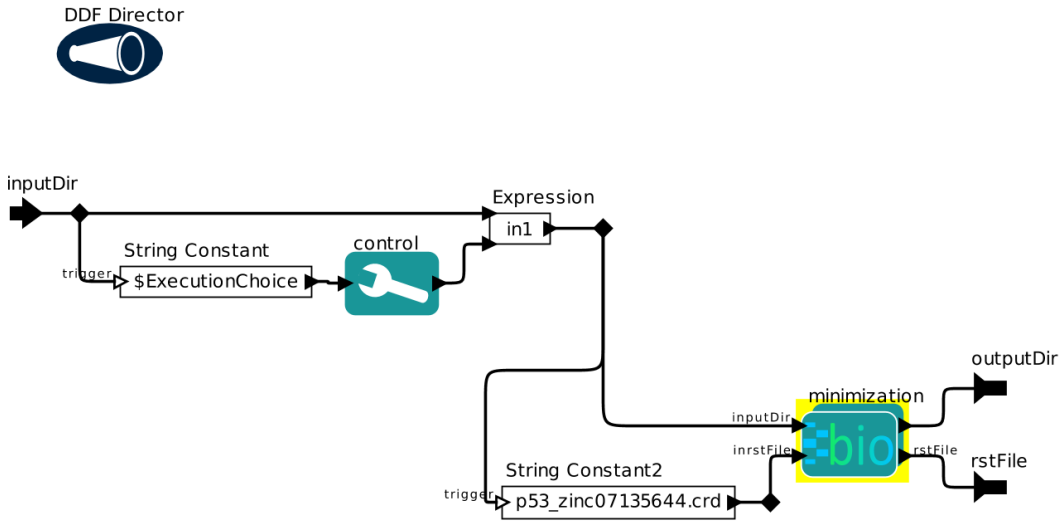
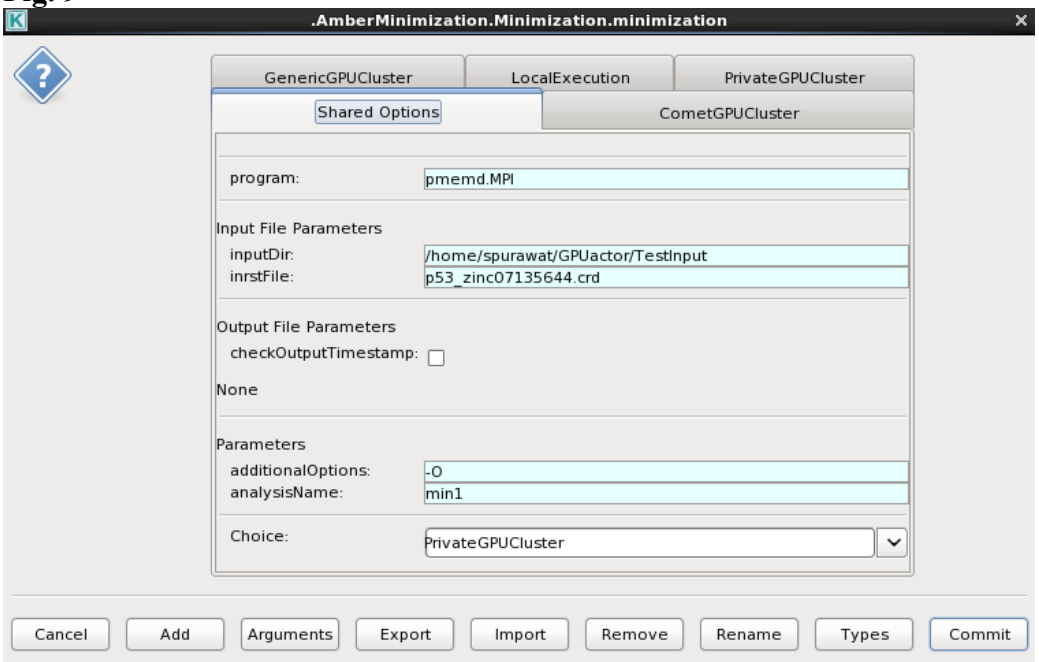
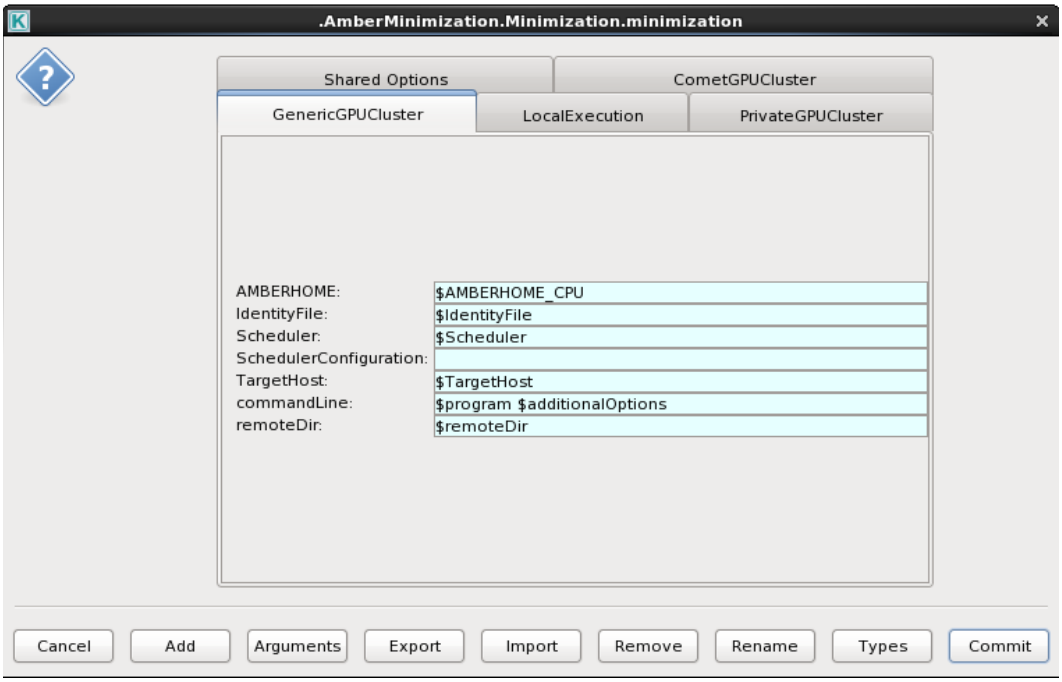


Fig. 9



Click GenericGPUCluster tab at the top of the screen (Fig. 9) and a list of parameters will appear (Fig. 10) There is one parameter (SchedulerConfiguration) that users must configure manually since it can be different for every simulation step.

Fig. 10



Here, we give an example of how to configure ScheduleConfiguration. ScheduleConfiguration takes the input from a submission script and feeds it into the scheduler. Below is an example submission script formatted for a SLURM scheduler (Fig. 11). To fill the variables for the ScheduleConfiguration parameter, the user types in the --name and the corresponding variables in one single line, leaving a space between each --name (Fig. 12).

Fig. 11

```
#!/bin/bash
#SBATCH --job-name="hellompi"
#SBATCH --output="hellompi.%j.%N.out"
#SBATCH --partition=compute
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=24
#SBATCH --export=ALL
#SBATCH -t 01:30:00

ibrun -v ../hello.mpi
```

Fig. 12

```
--job-name="hellompi" --output="hellompi.%j.%N.out" --partition=compute --nodes=2 --ntasks-per-node=24 --
export=All -t 01:30:00
```

Click Commit to accept the change.

Step 5.

Repeat steps 1 to 4

for the heating, equilibration, and production actors.

2. *Configuring Kepler for multiple workflow executions simultaneously through command line*

Step 1.

Close the Kepler GUI, if open

Go to KeplerData directory. It is usually located in the user's home directory

```
cd /home/username/KeplerData
```

or

```
cd ~/KeplerData
```

Step 2.

Edit DB Host parameter in configuration. file

```
vi modules/provenance/resources/configurations(latest)/configuration.
```

In configuration file: provide DB Host value as "localhost" and save changes.

```
<!-- host name running the database -->
<pair>
    <name>DB Host</name>
    <value>localhost</value>
</pair>
```

Step 3.

Start HSQL

Go to the directory where Kepler is installed. Start HSQL.

```
cd /PATH/WHERE/KEPLER/IS/INSTALLED/kepler-2.5
```

```
kepler.sh -hsql start
```

Step 4.

Run multiple workflows through the command line

```
./kepler.sh -runwf -nogui -parameters values /PATH/workflow1.
./kepler.sh -runwf -nogui -parameters values /PATH/workflow2.
./kepler.sh -runwf -nogui -parameters values /PATH/workflow3.
./kepler.sh -runwf -nogui -parameters values /PATH/workflow4.
```

Step 5.

Stop HSQL when you are done

```
kepler.sh -hsql stop
```

3. *Testing the reproducibility of the workflow by replacing random number seeds*

After running the workflow, you may want to check that the results from the workflow are reproducible; that is, for a given input, the same output is produced in additional runs of the workflow. To do so, you will need to modify the usrConf actors for heating, equilibration, and production in the usrConfMod actor in the workflow. You can do this through the Kepler GUI. You will also need the **MDCADD-Report text file (e.g., MDCADD-Report.txt)** generated from the run whose results you want to reproduce.

Instructions for configuring random number seeds directly from the command line are given in step 4, but it is recommended that you read steps 1-3 first.

Step 1.

Modify Heat_usrConf in the usrConfMod actor in the Kepler GUI

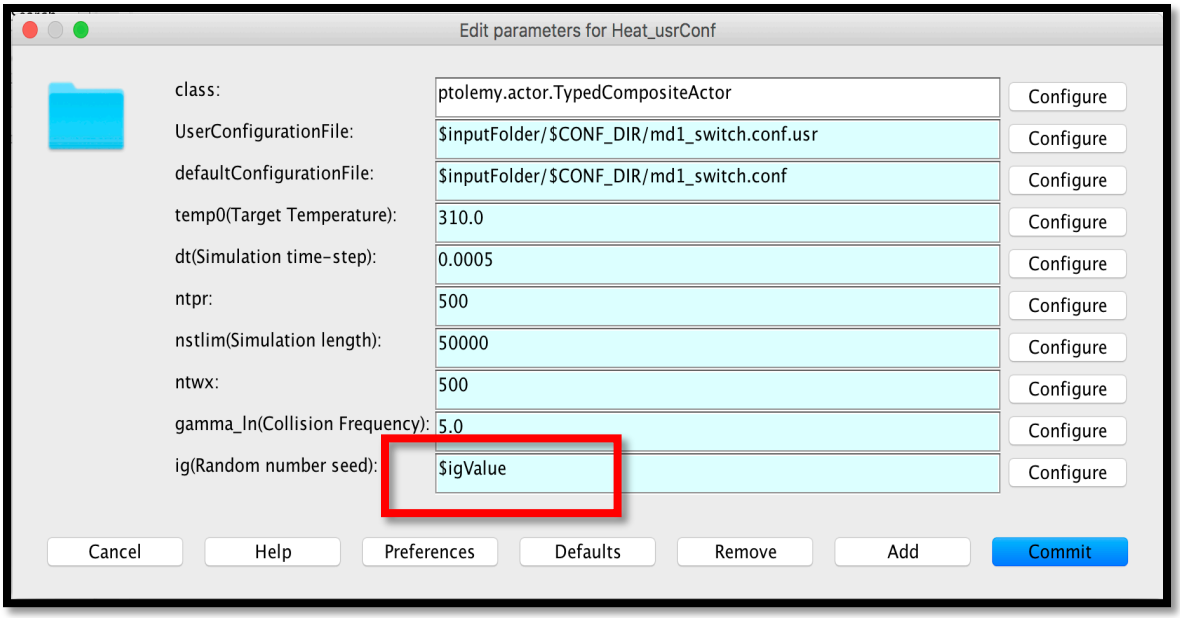
After opening the workflow in the Kepler GUI, right-click or control-click to open the "**usrConfMod"** actor and double click on the "Heat_usrConf" actor. You should see a window, with the heading "Edit parameters for Heat_usrConf," open (see Fig. 13a).

Substitute **$igValue** (the random number seed) with the value from the MDCADD-Report. For example: The MDCADD report includes:

"Random Number Seed used in Heating simulation: 447998"

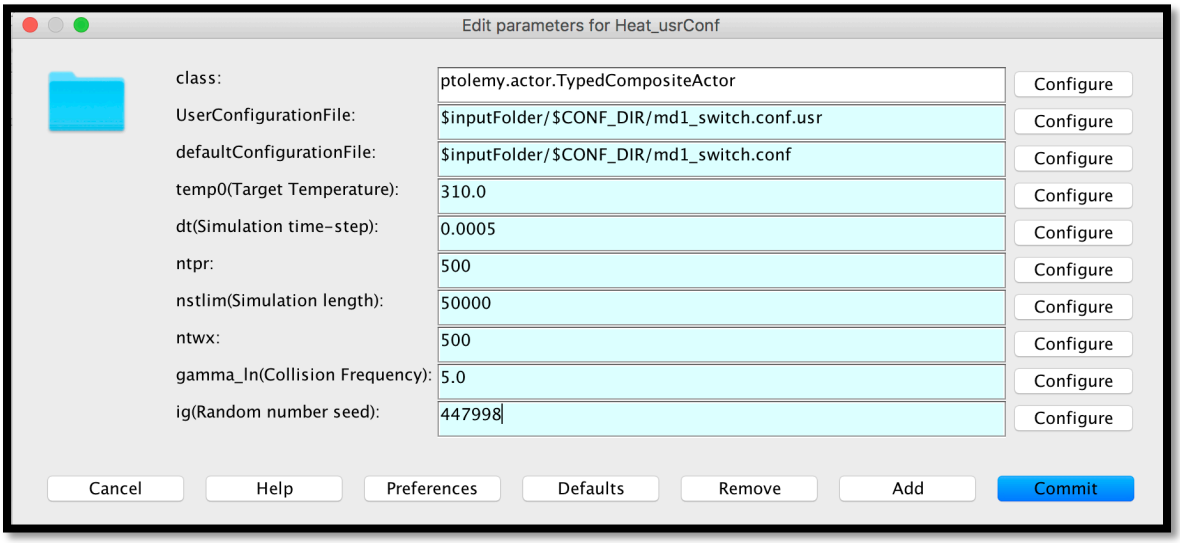
Replace $igValue with 447998

Fig. 13a


Edit parameters for Heat_usrConf
class:
ptolemy.actor.TypedCompositeActor
Configure
UserConfigurationFile:
$inputFolder/$CONF_DIR/md1_switch.conf.usr
Configure
defaultConfigurationFile:
$inputFolder/$CONF_DIR/md1_switch.conf
Configure
temp0(Target Temperature):
310.0
Configure
dt(Simulation time-step):
0.0005
Configure
ntpr:
500
Configure
nstlim(Simulation length):
50000
Configure
ntwx:
500
Configure
gamma_ln(Collision Frequency):
5.0
Configure
ig(Random number seed):
$igValue
Configure
Cancel
Help
Preferences
Defaults
Remove
Add
Commit

After substitution:

Fig. 13b



Select Commit and close the edit parameters window.

Step 2.

Modify the three Equili_usrConf actors in usrConfMod

Repeat step 1 for Equili1_usrConf, Equili2_usrConf and Equili3_usrConf with their respective random seed values for Equilibration1, Equilibration2 and Equilibration3 from the MDCADD-Report.

Step 3.

Modify the Prod_usrConf actor in usrConfMod

Next, you will need to replace the random number seeds with the random number values in the Production user configuration actor.

In the "usrConfMod" actor, double-click on the "**Prod_usrConf**" actor. Substitute $igValue (the random number seed) with Production random number values, **separated by commas and without spaces**, from the MDCADD-Report.

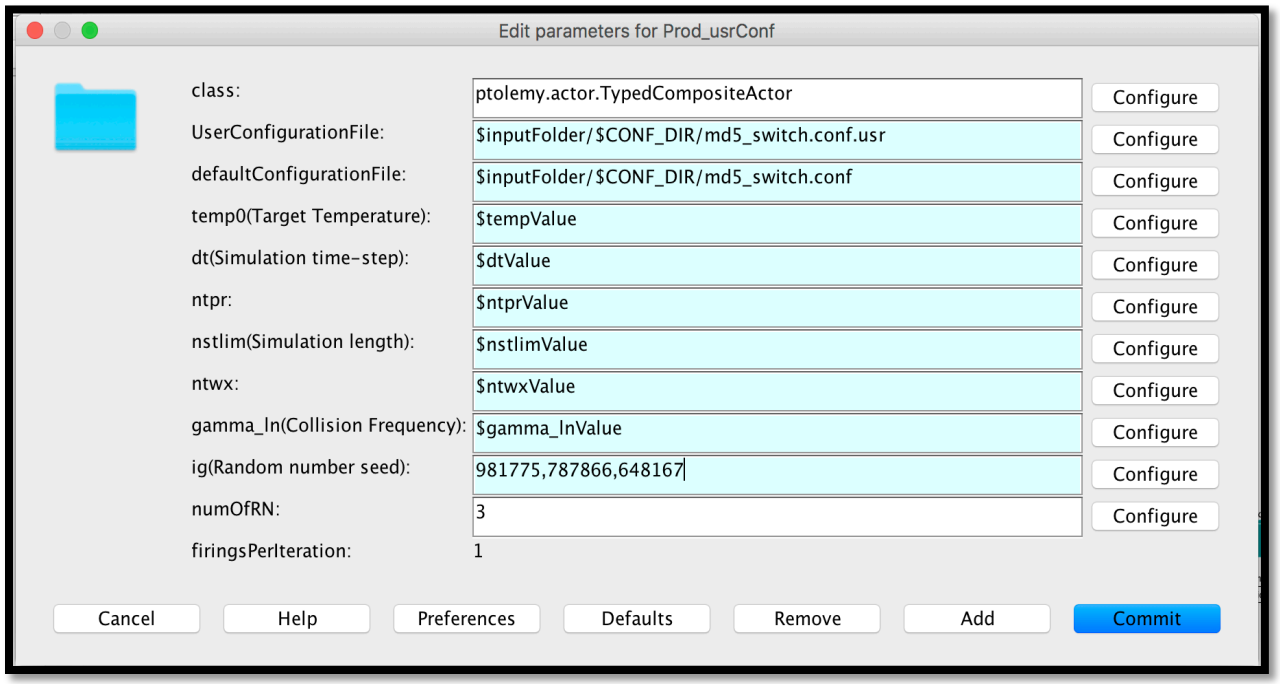
For example, MDCADD report includes:

Production1:
Production1 output folder:
/home/spurawat/GPUactor/TestInput/p53_zinc07135644/md5_Output1
Random Number Seed used in Production1 simulation: **981775**

Production2:
Production2 output folder:
/home/spurawat/GPUactor/TestInput/p53_zinc07135644/md5_Output2
Random Number Seed used in Production2 simulation: **787866**

Production3:
Production3 output folder:
/home/spurawat/GPUactor/TestInput/p53_zinc07135644/md5_Output3
Random Number Seed used in Production3 simulation: **648167**

Fig. 14.



Select Commit and close the edit parameters window. You can now test the reproducibility of your workflow.

Step 4.

Configure random number seeds with a command line switch

You can also configure the random number seeds when starting the workflow in the the command line. To configure the random numbers for the Heating, three Equilibration, and Production user configurations (usrConf), you will need to incorporate –usrConfMod switches for each stage.

This switches will require the appropriate random numbers. These numbers can be found in the workflow-generated MDCADD report.

Examples of the switches, with random numbers from the MDCADD report, are given below:

```
-usrConfMod.Heat_usrConf.igValue 4477998
 ...
-usrConfMod.Prod_usrConf.igValue 981775,787866,648167
```

NOTE: For Production, commas and no spaces separate the random number values. Maintain the order of your production jobs: Production1,Production2,Production3.

Incorporate the switch into your command to run the workflow, as shown below:

```
./kepler.sh -runwf -nogui –usrConfMod.Heat_usrConf.igValue
4477998 ... -usrConfMod.Prod_usrConf.igValue
981775,787866,648167 -inputFolder /home/shpurawat/TestInput -
nstlimValue 5000000 -numOfJobs 3 -ExecutionChoice LocalExecution
/home/shpurawat/Final_WF_Folder/MDCADD-RN
2016/MDCADD_WF_SingleCmpd-local-Reprod.xml
```

Appendix 1

My workflow doesn't run. Why?

There could be many reasons. The fastest way is to identify the error messages listed in the codes generated after starting the workflow. Here are some examples:

It said " [null] Command failed." What now?

This error means when Kepler failed on executing one of command prompt that run the simulation jobs. You should look further down to see which command it's failing on.

It said " ptolemy.kernel.util.IllegalActionException: Executing command ""mpirun -v -np 18 /soft/amber/latest/AmberTools/bin/pmemd.MPI -O -i /p53_tests/confDir/min1_switch.conf.usr -o /p53_tests/p53_wt/min1.out -p /p53_tests/p53_wt/p53_wt.top -c /p53_tests/p53_wt/p53_wt.crd -r /p53_tests/p53_wt/min1.rst -ref /p53_tests/p53_wt/p53_wt.crd"" returned a non-zero return value of 1." What does that mean?

This is the command for the first step of minimization as shown by this line "-o /p53_tests/p53_wt/min1.out". This error message means that Kepler has thrown an exception at minimization1 and could not continue to run the simulations. As this error is thrown at the beginning of running the workflow, it could possibly mean that the environment variables require to run the workflow is not set up properly. The environment variables could include but not limited to for example$AMBERHOME, MPI library and Amber libraries, etc.

It said "[null] mpirun detected that one or more processes exited with non-zero status, thus causing the job to be terminated. The first process to do so was: Process name: [[7877,1],0]" What's going on?

If this message is before or after an error message and the workflow stop responding, it means that the MPI library is not set correctly or the computer is capable of running MPI processors. When running minimizations using multiple CPUs, Amber called MPI installed and pre-set in user's local machine. If Amber cannot find MPI through user's environment, Kepler will throw the exception above.

If I don't set up the SSH key, can I still run the workflow with the GPUClusterJobSubmission?

Ideally, yes. You can still run the workflow using GUI if you don't not set up the SSH key. The graph interface will pop up a window asking for password when it need to connect to a remote cluster. However, the process will fail if the workflow is running with the command prompt. If a user is using the LocalExecution choice, it is not necessary to set up a SSH key.

It said "Unable to start application. java.lang.Exception: Failed to parse……" What's wrong?

Double check the command line that was typed to start the workflow. More likely than not, there is a misspell somewhere along the long list of command. This error message will also show up if complete path of a directory is not typed in. For example, instead of /home/username/input/, users replace /home/username with ~ as in in ~/input, the workflow will also fail with the error message shown above.

References

1. Karplus M & McCammon JA (2002) Molecular dynamics simulations of biomolecules. *Nat Struct Biol* 9(9):646-652.
2. D.A. Case JTB, R.M. Betz, D.S. Cerutti, T.E. Cheatham, III, T.A Darden, R.E. Duke, T.J. Giese, H. Gohlke, A.W. Goetz, N. Homeyer, S. Izadi, P. Janowski, J. Kaus, A. Kovalenko, T.S. Lee, S. LeGrand, P. Li, T. Luchko, R. Luo, B. Madej, K.M. Merz, G. Monard, P. Needham, H. Nguyen, H.T. Ngyuen, I. Omelyan, A. Onufriew, D.R. Roe, A. Roitberg, R. Salomon-Ferrer, C.L. Simmerling, W. Smith, J. Swails, R.C. Walker, J. Wang, R.M. Wolf, X. Wu, D.M. York and P.A. Kollman (2015).
3. Altintas I*, et al.* (2004) Kepler: An extensible system for design and execution of scientific workflows. *16th International Conference on Scientific and Statistical Database Management, Proceedings*:423-424.
4. Ilkay Altinas JW, Daniel Craw, Weizhong Li (2012) Challenges and Approaches for Distributed Workflow-Driven Analysis of Large-Scale Biological Data. in *EDBT Extending Database Technology*, ed Divesh Srivastava IA (ACM), pp 73-78.
5. Team RC (2015) R: A language and environment for statistical computing. (R Foundation for Statistical Computing, Vienna, Austria).
6. Roe DR & Cheatham TE, 3rd (2013) PTRAJ and CPPTRAJ: Software for Processing and Analysis of Molecular Dynamics Trajectory Data. *J Chem Theory Comput* 9(7):3084-3095.
7. Humphrey W, Dalke A, & Schulten K (1996) VMD: Visual molecular dynamics. *J Mol Graph Model* 14(1):33-38.
8. Pettersen EF*, et al.* (2004) UCSF chimera - A visualization system for exploratory research and analysis. *J Comput Chem* 25(13):1605-1612.