Installing AMReX and dependencies

Step 1: Create a local folder named AMREX to consolidate everything related to AMReX (E.g. users can also explore amrex-tutorials by downloading it in this folder).

Step 2: Type the following in the terminal "**sudo apt get update**"

Step 3: Type the following in the terminal "**sudo apt install git**"

Step 4: Visit AMReX-Codes · GitHub, go to amrex repository and copy the URL to clone amrex.

Step 5: Navigate to the AMREX folder, and inside that folder, type the following "**git clone <paste the clone URL for amrex here>**"

Step 6: Visit MicroSim, copy the URL to clone Microsim.

Step 7: Clone the Microsim repository outside the AMREX folder by typing the following "**git clone <paste the clone URL for microsim here>**"

Step 8: After getting Microsim, type the following in the terminal "**sudo apt install make**"

Step 9: Now we need to update the gcc on the machine, type the following in the terminal "**sudo apt build-essential**"

Step 10: AMReX needs both C and Fortran compilers. To get the FORTRAN compiler, type the following in the terminal "**sudo apt install g-fortran**"

Step 11: AMReX needs an MPI wrapper, to get mpich (MPI wrapper), type the following in the terminal "**sudo apt install mpich**"

Step 12: To run the Grand Potential code, one need the gsl library, to get the library, type the following in the terminal "**sudo apt install libgsl-dev**"

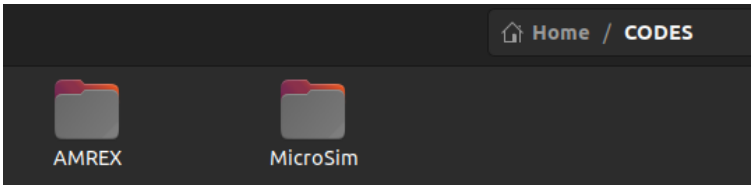
Step 13: Once gsl is available on the system, type the following in the terminal "**which gsl-config**". This will give the path of the gsl library.

```
nasir@nasir-VirtualBox:~$ which gsl-config
/usr/bin/gsl-config
nasir@nasir-VirtualBox:~$
```

Here, navigate out of **/bin** to find all the files located in **/usr.** Copy the path of the include and lib folder, which will be required later.

```
nasir@nasir-VirtualBox:~$ which gsl-config
/usr/bin/gsl-config
nasir@nasir-VirtualBox:~$ cd /usr
nasir@nasir-VirtualBox:/usr$ ls
bin    include  lib32  libexec  local  share
games  lib      lib64  libx32   sbin   src
nasir@nasir-VirtualBox:/usr$
```

Step 14: Go to the Grand_potential_AMReX folder present in the Microsim folder. Go to Exec, and open the GNUmakefile. Update the library path of gsl in the LIBRARY_LOCATIONS. Add the path to include files in the INCLUDE_LOCATIONS. The AMREX_HOME is the location of the amrex folder on your local system. Currently, the path to amrex home is set as ../../../AMREX/amrex. This will work only if AMREX and Microsim are installed exactly as shown below, or else the user will have to edit the path accordingly.



```
# AMREX_HOME defines the directory in which we will find all the AMReX code.
AMREX_HOME ?=/home/nasir/AMREX/amrex
LIBRARY_LOCATIONS += /usr/lib
XTRALIBS += -lgsl -lgslcblas -lm

DEBUG        = FALSE
USE_MPI      = TRUE
USE_CUDA     = FALSE
USE_OMP      = FALSE
COMP         = gnu
DIM          = 2

include $(AMREX_HOME)/Tools/GNUMake/Make.defs

include ../Source/Make.package
VPATH_LOCATIONS  += ../Source
INCLUDE_LOCATIONS += ./Source /usr/include

include $(AMREX_HOME)/Src/Base/Make.package

include $(AMREX_HOME)/Tools/GNUMake/Make.rules
```

Changes required

Step-16: Navigate to the Exec folder in Grand_potential_AMReX, and open the **bash.sh**. Give the number of processors required to carry the task in the highlighted position. If the user wants to run the code serially, then mention the number of procs as 1. Note that the name of the executable depends on which flags are set to TRUE in the GNUmakefile. In the below example, as CUDA was true the executable name contains CUDA as well.

```
#!/bin/bash

make

g++ -o Replace Replace.cpp

./Replace

mpirun -np 64 ./main2d.gnu.MPI.CUDA.ex input2.in
```

Step 17: Now AMReX is Ready to run. To run the code, navigate to the Exec folder present in Grand_potential_AMReX in the terminal. Type the following in the terminal "**./bash.sh**". This will first build the executable, make a relevant input file and will later run on the number of procs as defined previously.

Step-18: The plot files will be generated, which can be later viewed in any post-processing software.