Time - evolving COCONUT manual

By Michaela Brchnelova, Haopeng Wang, Yucong Li, Centre for Mathematical Plasma Astrophysics

,

The detail description of the first time-evolving COCONUT is available in [1]. The necessary scripts are contained in the folder titled "te_coconut", available at:
`https://kuleuven-my.sharepoint.com/:f:/g/personal/haopeng_wang1_kuleuven_be/ErQPF-9dGMZDsly7s9dorYIBCNX78HKZY9ZZFL8gl493rw?e=Ggw76J`.

For the latest version described in [2]. The necessary scripts are also available via the same link, but in the folder titled "TestCaseAround2024MayEvent(TheLatesVersion)". Several files across six directories—located within the "Plugins" and "src" folders—have been modified. These files are provided in the folder "DATA_DiskSection". Each directory contains all relevant files, including both modified and unmodified ones, along with a subfolder titled "Files Modified for Time-Evolving Simulations", which includes only the modified files and a corresponding "README" file.

The CFcase, magnetograms and grid meshes are included in the folder "Scratch_DiskSection". This test case is part of the simulations described in [2]. It reads 121 hourly-updated magnetograms at the start of the time-evolving coronal simulation, and is configured to run on the Level-5 mesh, the coarsest resolution, and simulates one day of coronal evolution in physical time. You may adjust the simulation duration by modifying the parameter "Simulator.SubSystem.MaxTime.maxTime" in the "map_unsteadyeclipse_fullMHD.CFcase" file, provided it remains less than 120.0*3600.0/1447.2. Alternatively, Level-6 or Level-7 meshes can be selected by editing the CFcase files. We recommend using the Level-6 mesh, as it provides a well-balanced compromise between computational efficiency, accuracy, memory usage, and numerical stability. For Level-6, it is advisable to use 900 CPU cores instead of the 270 configured for Level-5 mesh.

Please also read the two README files located in the parent directories "DATA_DiskSection" and "Scratch_DiskSection" before executing this test case. Additionally, we have shared a complete COOLFluiD directory, except for the folder OPENMPI, in folder titled "Complete-COOLFluiDdirectorywithoutOPENMPI(2024MayEvent)".

1. Installation of base COCONUT

In this part you will be guided to install a 'base version' of COCONUT on VSC. This part was developed by Michaela.

Just to be on the same page when describing the procedures below:

-I don't know what your base COOLFluiD directory is called, everyone has their own, so here I just call it "COOLFluiD". You have to replace that with whatever you specifically have, e.g. "COOLFluid Genius".

-Your source code, so all the C++ files etc., is in COOLFluiD / plugins / *, e.g. COOLFluiD / plugins / FiniteVolumeMHD.

-Your setup files (CFcase, CFmesh, etc.) that you actually run to generate results are stored in COOLFluiD/OPENMPI/optim/plugins/*, for example COOLFluiD / OPENMPI / optim / plugins / MHD / testcases / SolarCorona etc.. For some of you the "optim" might be "release" or so, but again, I mean the same thing, so just insert whatever you have.

Now, you have to decide what you want to do. The option A is to do this re-installation by re-pulling the code from Github, just like when you installed it for the first time, with some modifications. However, that means you would lose all the changes you have made to the source code (COOLFluiD/plugins/*) in your own installation (so you would need to put this somewhere else and then place it back, recompile and pray). If you want to do this, as it is simpler, follow instructions A. If you do not want to re-sychronise with Github and keep the changes you have made to the source code, follow instructions B.

1.1. A. Re-installing via synchronisation with Github

One more warning before your proceed. This approach will change your source code, so all your own code in the COOLFluiD/plugins/* destination, and any other changes you have made to COOLFluiD. It will also erase all your setups, in COOLFluiD/OPENMPI/optim/plugins/* or COOLFluiD / OPENMPI / release / plugins /* (depending on what you are using), but both approaches will do that. You can back up your current source code and setups (see B.3 for options how this can be done) beforehand somewhere else if you do not want to lose it. Backing it up on SCRATCH might be easier since there is more space for that, but keep in mind that it will be erased in 28 days, so do not leave it there for too long.

Here, we follow the instructions online with some modifications:

1.1.1 A.1.

The first step tells you to get the "install_COOLFluiD.sh" from Github, but we do not do that as it has the 2018 paths in it. So instead, you use the file "install_COOLFluiD.sh" that I attached, which has the new versions. You save this in the DATA partition where you want to re-install COOLFluiD.

1.1.2 A.2.

Like the website says, in this install file, change the TOP_DIR and COOLFLUID_TOP_DIR to the destination of the installation - just like you did when you did this for the first time.

1.1.3 A.3.

When the install script is modified, run (in the same location):

./install_COOLFluiD.sh DEBUG_NOCUDA –download=2

Try the following command if you had permission denied in A.3:

chomd -R 777 /your working path

optinal:

chmod a+x ./install_COOLFluiD.sh

chmod u+x ./install_COOLFluiD.sh

1.1.4 A.4.

The above command will start first downloading all the files (from Github) and then installing them. The downloading part should go OK (as long as you have enough disk space), but the installation will fail. That's ok. (Actually, the installation should also work for the current version, you should neglect A.5. and A.6.)

1.1.5 A.5.

Once the installation has failed, you download the "COOLFluid_Genius_nocuda.conf" file and change the TOP_DIR and COOLFLUID_TOP_DIR in it as well, same as before. Then, use it to replace the configuration file with the same name in TWO places (not one, two): in your base COOLFLuiD directory (COOLFluiD/) and in the Genius configuration COOLFLuiD directory, (COOLFluiD/tools/conf/Genius/).

1.1.6 A.6

Now, you run

./install_COOLFluiD.sh DEBUG_NOCUDA –download=1

and this should not crash. It will take a few hours to finish. If it crashes, you contact me and we cry together (joking, if it crashes, double check that you have correctly replaced the configuration file and installation files at the right places with the ones attached here first. If yes and it still crashes, then you let me know).

1.1.7 A.7.

Once the installation is finished, you put all your stuff back in (so, the backed up source code and setups) at the right places (so COOLFluiD/plugins/ for source code and COOLFluiD/OPENMPI/optim/plugins/ for setups) and recompile and hope that it will be fine. I attached my recompilation script as "compilescript.sh" which contain the new modules - remember that inside, you have to adjust the paths to those that fit for you (again, the TOP_DIR and COOLFLUID_TOP_DIR). I usually have and launch this script from the OPENMPI/ location using "sh".

1.1.8 A.8.

Now, to actually run your cases, all the pbs/ slurm scripts that you use to submit jobs will have to have the new modules in them instead of the old ones + a new line in the beginning, so these are:

```
module load cluster/genius/batch
module load CMake/3.20.1-GCCcore-10.3.0
module load Boost/1.76.0-GCC-10.3.0
module load ParMETIS/4.0.3-gompi-2021a
module load PETSc/3.15.1-foss-2021a
```

and with that, the new CF should run (so just erase the old module load lines and replace them with these ones). If it does not, let me know.

1.2. B. Re-installing without synchronisation with Github

This approach will not change your source code.

1.2.1 B.1.

Check if you last pulled your COOLFluiD version from Github before or after November 4, 2022. If it is after, proceed with the next steps. If it was before, let me know, there are a couple of updates to your plugins that will have to be made first, but I will tell you personally because I am not sure for how many of you this is the case.

1.2.2 B.2.

To re-install the code without any paths to the 2018 toolchain, you need the "COOLFluid_Genius_nocuda.conf" file that is attached to this email. Download this configuration file, change the TOPTo re-install the code without any paths to the 2018 toolchain, you need the "COOLFluid_Genius_nocuda.conf" file that is attached to this email. Download this configuration file, change the TOP_DIR and COOLFLUID_TOP_DIR in it to the appropriate paths, and replace the one with that same name that you had in two places (not one!): your base COOLFluiD directory (COOLFluiD/ or whatever name you have for it) and also the one with the same name in the folder COOLFluiD/tools/conf/Genius/ . It is important that it is this one that is being used for re-installation since this one does not contain the paths to the 2018 libraries. DIR and COOLFLUID_TOP_DIR in it to the appropriate paths, and replace the one with that same name that you had in two places (not one!): your base COOLFluiD directory (COOLFluiD/ or whatever name you have for it) and also the one with the same name in the folder COOLFluiD/tools/conf/Genius/. It is important that it is this one that is being used for re-installation since this one does not contain the paths to the 2018 libraries.

We do the same with the "install_COOLFLuiD.sh" file that is in your DATA directory (or wherever you store COOLFluiD) - use the one attached to this email, change the TOP_DIR and COOLFLUID_TOP_DIR to the appropriate paths, and replace the one that you have there now.

1.2.3 B.3.

This process will rewrite your COOLFluiD/OPENMPI/optim/plugins/* files (or some of your have COOLFluiD/OPTIM/release/plugins/*) - so, basically where you have your setups and some of your results (so they will be gone). It is thus important that you BACK THESE UP or MOVE THEM AWAY so they are not touched by the re-installation process, but that's not hard at all.

You can either just copy the entire COOLFluiD/OPENMPI/optim/plugins/ subfolder (or the subsequent subfolders, depends on what you are using) to a different location using cp -r [directory] [final path], or you can zip it using tar -zcvf [filename.tar.gz] [directory] and move it away with a smaller size, or you can simply move the entire directory to a different location via mv [directory] [final path], outside of CF so that it is not touched by the re-installation (but then you will not have a backup if something goes wrong). The last option is the fastest.

Keep in mind that your DATA might become too full with all of these backups and later lead to installation failures. Thus, perhaps use SCRATCH for these backups, but keep in mind that stuff gets cleaned away from SCRATCH after 28 days, so you have to finish this by then.

1.2.4 B.4.

This process WILL NOT rewrite your source code plugins in COOLFluiD/plugins/ , which is why we are doing this and not process A. However, shit can still hit the fan (sorry Fan) and I would still suggest that you back these files up anyway (or at least the relevant subdirectories you are using) just like in step B.3., because it is easy to make a mistake and somehow still remove them by accident (e.g., changing "1" to "2" in the process below will do that). Backups are always recommended. So you can cp -r your COOLFluiD/plugins/ to e.g. SCRATCH just in case.

1.2.5 B.5.

Once you have everything that you need backed up, and the configuration and install files are replaced, you run:

./install_COOLFluiD.sh DEBUG_NOCUDA –download=1

and wait until the process is (hopefully) successfully finished successfully - this will be hours. Beware of the –download=1 option. Make sure you type "1" because if you type "2", automatically things will start getting downloaded and replaced by stuff from Github, which we do not want.

1.2.6 B.6

After the re-installation is done, put your backed-up setups back into COOLFluiD/OPENMPI/optim/plugins/ or wherever you had them before.

1.2.7 B.7

Your code should now be the same one as you had before from user's perspective. Still, double-check that all your original modifications in COOLFluiD/plugins are still there. If you make additional changes to the source code, you can use the recompilation script attached to this email, "compilescript.sh", in which you have the right modules. Here, you have to again set your own TOP_DIR and COOLFLUID_TOP_DIR. I usually launched this recompilation script using "sh" in OPENMPI/.

1.2.8 B.8

Your pbs/ slurm scripts to submit the jobs will now require to have the new modules in them instead, see step A.8.

2. Running of the Time-Evolving COCONUT

The detail description of the time-evolving COCONUT is available in [1]. The necessary scripts are available online[1].

You can also run it as in the previous COCONUT version, where the inner boundary field is calculated by the PF solver and does not evolve over time, by commenting out the following line in the CFcase file:

"Simulator.SubSystem.Flow.Jet1.VarIDs = 0"

There will be some discrepancies in the inner-boundary magnetic field calculated from the same magnetogram by the previous COCONUT version and the current version where the inner-boundary magnetic field is directly interpolated from the magnetogram '.dat' file.

[1]https://kuleuven-my.sharepoint.com/:f:/g/personal/haopeng_wang1_kuleuven_be/Eq09876phBFImiuJeo8UWD8B8yRaBlxuWnx528NAwbNGGQ?e=Vn6V1M

2.1. Preparation

First you have to replace some file attached in the given folder ‘script’. You need to find the path where you installed COCONUT and replace these files in the corresponding folder. You should open the already replaced files, add a space in any blank area, and then save them. This will ensure the replaced files are recognized as modified and will be recompiled.

There is a user-defined parameter, ‘maxvA", in the files ‘StdUpdateSolPP.cxx", ‘UpdateSolMHD.cxx" and ‘SuperInletProjectionConstrained.cxx". It is suggested to set it as ‘CFreal maxvA = 3.0e6;" for solar maximum conditions and ‘CFreal maxvA = 2.0e6;" for solar minimum conditions. For the latest version, it is defined and adjusted automatically in StdUpdateSolPP.cxx", ‘UpdateSolMHD.cxx", but still not yet for ‘SuperInletProjectionConstrained.cxx".

You can put your downloaded magnetograms under the folder ‘MapData’. The python script “coconut_bcfile-3.py" can be used to create a series of magnetogram data files. “candence" refers to the time interval between two adjacent magnetograms. You should modify “days_Feb" to reflect the days of February for the corresponding year. In “Frame_num = 31*hour_ave + hour_ave", the value "31" is an adjustable parameter which represents the number of days you need. And that determines how many magnetograms will be generated in a single execution. For now it is not encouraged to simulate years with different days in Feb. at the same time. But if your simulation ends before the last day of Feb. then it’s OK.

The python script “coconut_bcfile-3.py" may crash. If that happens, you can restart it by modifying “ii_begin" to the value of the last “ii-hour_ave" and adding “hour_ave"+1. The value of “ii-hour_ave" is updated and displayed in sync with the code execution..

Before submitting a job to the VSC HPC, you should create a symbolic link to the path of the xxx.slurm file using the command like this:

ln -sf /data/leuven/357/vsc35794/COOLFluiD_Genius_New/OPENMPI/optim/apps/Solver/coolfluid-solver* ./

(We usually install COCONUT to data partition and submite a job from scratch partition, also save the output files in this partition.)

2.2. Complie

You can find a file named **NodalStatesExtrapolator.ci** in the path of .../the path you downloaded COCONUT/src/Framework. In the file, you need to change a few lines concerning to ‘case dependent 1-5’.

1.Case-dependent_1 is determined by the structure of the file names of your magnetograms. For example, for a series of data named map_adapt_lmax15_20070909020000.dat. case-dependent_1 will be like this:

std::string FileName_prefix="./MapData/map_adapt_lmax15_2007";

std::string FileName_suffix="0000.dat";

2. In case-dependent_2, you need to specify these two following parameters. “CFuint m_magfiles" is the number of magnetograms you provided, corresponding to $t_{-1}, t_0, ...t_n, t_{n+1}$, where the simulation covers the time interval between t_0 and t_n. And “CFuint Cadence" is the time interval between two adjacent magnetograms (in hours).

3.In case-dependent_3, you can change the year of this simulation and days of February of this year. “CFuint year=2007" and “CFuint days_Feb=29"

4.For case-dependent_4, “CFuint data_start" is the time for starting simulation t_0, and the stucture will be like this:

CFuint data_start=2007091014 for September 10 at 14 O’clock

CFuint data_start=2007101014 for October 10 at 14 O'clock

5.In case-dependent_5, you change "current_data" to the time t_{-1}, which is "Cadence" hours previous than t_0, and the structure is the same as case-dependent_4.

Then you can get access to the path /.../COOLFluiD_Genius_New/OPENMPI. Then you run: sh compilescript.sh

2.3. Run Time-Evolving COCONUT

To actually run the T-E COCONUT, there are 3 steps to go.

2.3.1 step 1

First, you can open the file named **map_steady.CFcase** and change the path of your data accordingly. The first part is in **Boundary conditions**. The correct one will be like this:

Simulator.SubSystem.EM.Data.DistanceBased.FileNameTw =./MapData/map_adapt_l-max15_20070910140000.dat The time of this magnetogram is the first data in your studying carrington rotation, corresponding to t_0.

Then you change the file path of the Mesh reader part. Attention you have to mute the restart part. Simulator.SubSystem.CFmeshFileReader0.Data.FileName = ./30Rs_lvl5.CFmesh. the same for Simulator.SubSystem.CFmeshFileReader1.Data.FileName = ./30Rs_lvl5.CFmesh.

Another line to change is in the very bottom of the file and like this:

Simulator.SubSystem.Flow.Data.DistanceBased.FileNameTw = ./MapData/map_adapt_l-max15_20070910140000.dat

Then you will finish the running for step 1 by submitting **Quasi_steady.slurm**. Remember to change the path in this file accordingly.

2.3.2 step 2

After finishing step one, there will be a result file named **corona.CFmesh**. Now you open the file named **map_steady_fullMHD.CFcase** and change the lines of both first reader and second reader like this:

Simulator.SubSystem.CFmeshFileReader0.Data.FileName = ./result/corona.CFmesh. The same for Simulator.SubSystem.CFmeshFileReader1.Data.FileName

Attention this time you will need the Restart part and mute the lines in the step 1.

Again there is another line to change which is in the very bottom of the file and looks like this:

Simulator.SubSystem.Flow.Data.DistanceBased.FileNameTw = ./MapData/map_adapt_l-max15_20070910140000.dat. The same for Simulator.SubSystem.EM.Data.DistanceBased.FileNameTw

Now you can submit the **Quasi_steady_fullMhd.slurm**. Remember to change the path in this file accordingly.

2.3.3 step 3

Now you can open the file named map_unsteadyeclipse_fullMHD.CFcase and change the path accordingly in the same way as step 2. Remember to modify this line if you have customized filename.

Simulator.SubSystem.Tecplot1.FileName = corona_cr2062.plt

Now for running T-E version, you will need 3 or more files in this line(it is located in the very bottom of this file)

# here you have to put the list of different magnetogram file names corresponding to different times already in the right format (You can also keep the current values unchanged, as they have already been defined within the COCONUT code, as described in Subsection 2.2.However you still need to put more than 3 files in here to activate time-evolving process.)

Simulator.SubSystem.Flow.Data.DistanceBased.FileNameTw = ./MapData/map_adapt_lmax15_20070910140000.dat ./MapData/map_adapt_lmax15_20070910080000.dat ./MapData/map_adapt_lmax15_20070910140000.dat ./MapData/map_adapt_lmax15_20070910200000.dat ./MapData/map_adapt_lmax15_20070911020000.dat

Simulator.SubSystem.Flow.Data.DistanceBased.FileNameTime = 0.0 -14.9792 0.0 14.9792 29.9584 (in code unite, calculated as physical hours * 3600.0/1447.2)

These files and times correspond to t_0, t_{-1}, t_0, t_1 and t_2.

#this is calculated by (cadence in hours)*3600.0/1442.0

Then you can submit the TimeevolvingfullMHD.slurm. During running, it is possible that the model would break due to memory loss or something like that. In that case, you would need to change filename in the CFcase file from 'corona.CFmesh' to the result of last CFmesh file before break. Additionally, you also need to check the time from the **slurm-xxxxx.outfile** before break and change the line of initial time like this:

Simulator.SubSystem.InitialTime = 0.0

# In code hour unite, the restart physical time *3600.0/1447.2, for first time running it should be set to 0.0

If the corresponding PhysicalTime is 1169.5009, it should be replaced as 1169.5009*3600.0/1447.2.

Also, ensure that "Simulator.SubSystem.MaxTime.maxTime"is set to at least $t_n \times \frac{3600.0}{1447.2}$. The code will terminate at this time.

Alternatively, such breakdown can be avoid by using the big memory nodes via.

#SBATCH –partition=bigmem

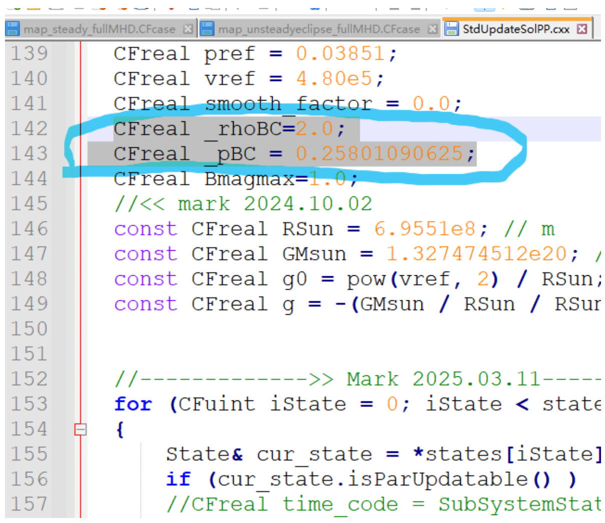
#SBATCH –mem-per-cpu=20000M (We found the execution time of the big memory node should be limited to within 12 hours. We suggest to use 4 nodes, 288 CPU cores for the "30Rs_lvl5.CFmesh")

Besides, you can reduce the number of CPU cores used per node and then allocate more memory per core by adding to your job script: '#SBATCH –mem-per-cpu=15000M"

The Fortran script "result_drawing_COCONUT_Seq.f90" can be used to creat a ".CFmesh" file from the corresponding ".plt" file. Keep in mind you should set "Time_sequence= .false." and "write_CFmesh = .true." Meanwhile, you should set "Seq_Start" to the integer hour of the time you want to restart from, then you run "make" to creak the "sperical.YY" file and run the code via "sbatch jobsub.slurm".

2.3.4 Numerical stability improvement

This should work for both time-evolving and queasi-steady-state coronal simulations. You can try to download the two file "StdUpdateSolPP.cxx" and "StdUpdateSolPP.hh" from "DATA_DiskSection/Plugins/NewtonMethod" directory to your own corresponding source code file, add "StdUpdateSolPP.cxx" and "StdUpdateSolPP.hh" to the "CMakeLists.txt", and recompile COCONUT. Remember to adjust "_rhoBC" and "_pBC" illustrated in Figure. 1 to the corresponding boundary value in your case.

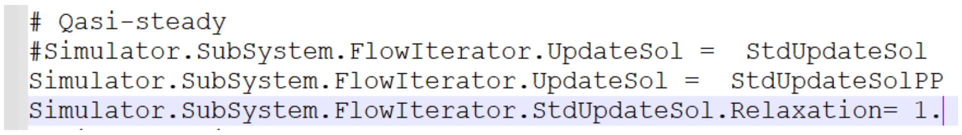


```
CFreal pref = 0.03851;
CFreal vref = 4.80e5;
CFreal smooth_factor = 0.0;
CFreal _rhoBC=2.0;
CFreal _pBC = 0.25801090625;
CFreal Bmagmax=1.0;
//<< mark 2024.10.02
const CFreal RSun = 6.9551e8; // m
const CFreal GMsun = 1.327474512e20; /
const CFreal g0 = pow(vref, 2) / RSun;
const CFreal g = -(GMsun / RSun / RSur


//------------->> Mark 2025.03.11-----
for (CFuint iState = 0; iState < state
{
    State& cur_state = *states[iState]
    if (cur_state.isParUpdatable() )
    //CFreal time_code = SubSystemStat
```

Figure 1: "_rhoBC" and "_pBC" are the plasma density and thermal pressure at the inner boundary.

Meanwhile, you should change or add "Simulator.SubSystem.FlowIterator.UpdateSol" in your CFcase as demonstrated in Figure. 2



```
# Qasi-steady
#Simulator.SubSystem.FlowIterator.UpdateSol =  StdUpdateSol
Simulator.SubSystem.FlowIterator.UpdateSol =  StdUpdateSolPP
Simulator.SubSystem.FlowIterator.StdUpdateSol.Relaxation= 1.
```

Figure 2: Implement the PP measure described in [2] to your simulation.

If it still crash, don't worry, we are developing simpler algorithms, but comparable in performance to the extended magnetic field decomposition approach [4] which do well in dealing with time-evolving problems with strong magnetic field, to further improve numerical stability of both time-evolving and quasi-steady-state COCONUT.

2.3.5 0.1AU Data

The Fortran script "result_drawing_COCONUT_Seq.f90" can also be used to extract data at 21.5 R_s for the machine learning project.

1. **Create "layerindex.dat"**. You should set "write_CFmesh = .false.", "SortorNot = .true." and keep the other parameters the same as in creating ".CFmesh" file. And then run the "result_drawing_COCONUT_Seq.f90" code. Remember to change the path in this following line:

filename='./corona_cr2061-time_//ctime//.plt' #This is the path and name of your previous simulation results.

Then you can search the file for 'filename' to modify the path and name of the 'layerindex.dat' and 'Radian.dat'.

2. **Create a series of the files at 21.5 R_s**. You should set "Time_sequence= .true.", "SortorNot = .false.", "pannels= .true." and "Zerod1AU_only= .true.". Keep in mind to modify "time_step=600" to the value of 100 times of the time interval (in unite of hour) between two adjacent results, and set "Seq_Start" and "Seq_End" to the time (in unite of hour) of

the first and last results. And then run the “result_drawing_COCONUT_Seq.f90" code. You can run multiple executions to reduce the time required for post-processing.

The radial basis function (RBF) method proposed in [3], which is numerically stable, spatially accurate, and efficient, is used in the “result_drawing_COCONUT_Seq.f90" code. Please kindly cite this paper for reference.

References

[1] H. P. Wang, S. Poedts, A. Lani, M. Brchnelova, T. Baratashvili, L. Linan, F. Zhang, D. W. Hou, and Y. H. Zhou. Efficient MHD modelling of the time-evolving corona by COCONUT. A & A, 694(A234), 2025.

[2] H. P. Wang, S. Poedts, A. Lani, L. Linan, T. Baratashvili, F. Zhang, D. Sorokina, J. H.-J., Y. C. Li, N.-Z. Mahdi, and B. Schmieder. Time-evolving coronal modelling of solar maximum around the May 2024 storm by COCONUT. 2025.

[3] H. P. Wang, C. Q. Xiang, X. J. Liu, J. K. Lv, and F. Shen. Implicit solar coronal magnetohydrodynamic (MHD) modeling with a low-dissipation hybridized AUSM-HLL Riemann solver. ApJ, 935(1):46, Aug 2022.

[4] H. P. Wang, L. P. Yang, S. Poedts, A. Lani, Y. H. Zhou, Y. H. Gao, L. Linan, J. K. Lv, T. Baratashvili, J. H. Guo, R. Lin, Z. Su, C. X. Li, M. Zhang, W. W. Wei, Y. Yang, Y. C. Li, X. Y. Ma, E. Husidic, J. H.-J., N.-Z. Mahdi, W. Wang, and B. Schmieder. SIP-IFVM: A time-evolving coronal model with an extended magnetic field decomposition strategy. ApJS, accepted, 2025.